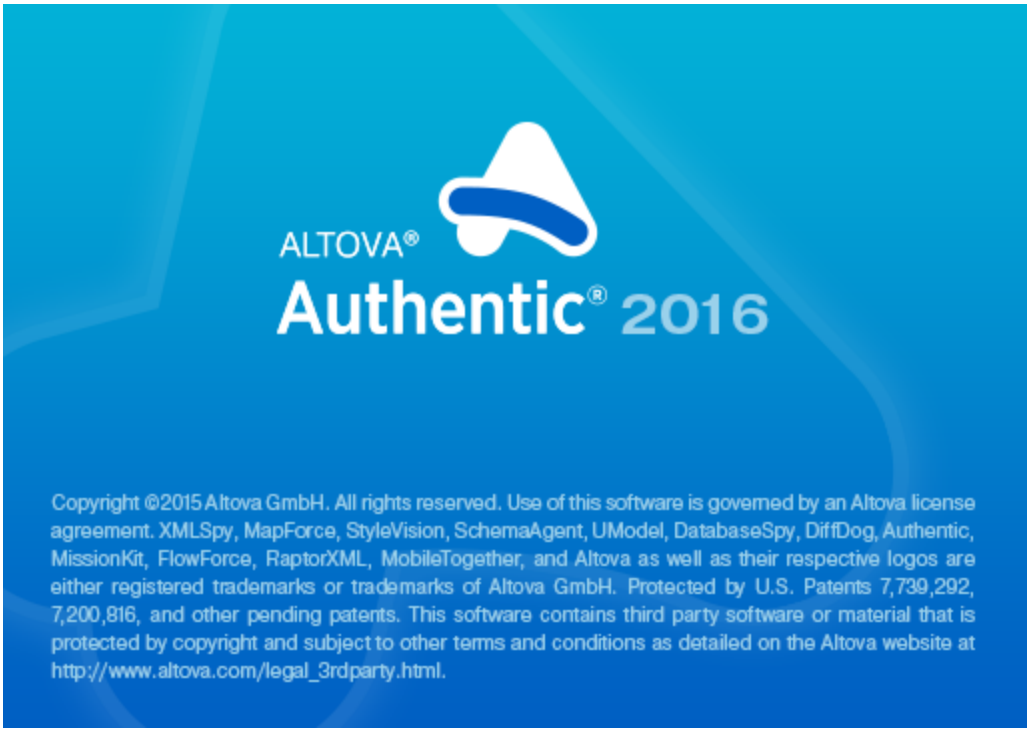
Manual del usuario y de referencia


ALTOVA®
Authentic® 2016
Copyright ©2015 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, MobileTogether, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html.

Manual del usuario y de referencia de Altova Authentic 2016 Community Edition

Todos los derechos reservados. Ningún fragmento de esta publicación podrá ser reproducido de manera alguna (ya sea de forma gráfica, electrónica o mecánica, fotocopiado, grabado o reproducido en sistemas de almacenamiento y recuperación de información) sin el consentimiento expreso por escrito de su autor/editor.

Los productos a los que se hace referencia en este documento pueden ser marcas registradas de sus respectivos propietarios. El autor y editor no afirman ser propietarios de dichas marcas registradas.

Durante la elaboración de este documento se tomaron todas las precauciones necesarias para prevenir errores. Sin embargo, el autor y editor no se responsabilizan de los errores u omisiones que pudiese contener el documento ni de los posibles daños o perjuicios derivados del uso del contenido de este documento o de los programas y código fuente que vengan con el documento. Bajo ninguna circunstancia se podrá considerar al autor y editor responsables de la pérdida de beneficios ni de cualquier otro daño y perjuicio derivado directa o indirectamente del uso de este documento.

Fecha de publicación: 2015

© 2015 Altova GmbH

Tabla de contenido

1 Altova Authentic 2016 Community Edition 2

2 Guía del usuario y de referencia 4

2.1 Interfaz y entorno ... 6
2.1.1 La interfaz gráfica de usuario (IGU) ... 7
- Ventana principal ... 9
- Ventana Proyecto ... 11
- Ventana Información ... 13
- Ayudantes de entrada ... 14
- Ventana de resultados: Mensajes ... 15
- Barra de menú, barras de herramientas y barra de estado ... 16

2.1.2 El entorno de aplicación ... 18
- Configuración y personalización ... 19
- Tutoriales, proyectos y ejemplos ... 20
- Características y archivos de ayuda ... 21

2.2 .Tutorial de la vista Authentic ... 22
2.2.1 Abrir un documento en la vista Authentic ... 24
2.2.2 La interfaz de la vista Authentic ... 26
2.2.3 Operaciones con nodos ... 29
2.2.4 Introducir datos en la vista Authentic ... 32
2.2.5 Introducir valores de atributo ... 35
2.2.6 Añadir entidades ... 36
2.2.7 Imprimir el documento ... 37

2.3 .Interfaz de la vista Authentic ... 38
2.3.1 La interfaz gráfica del usuario (IGU) ... 39
2.3.2 Iconos de la barra de herramientas de la vista Authentic ... 41
2.3.3 Ventana principal de la vista Authentic ... 44
2.3.4 Ayudantes de entrada de la vista Authentic ... 47
2.3.5 Menús contextuales de la vista Authentic ... 52

2.4 .Edición en la vista Authentic ... 55
2.4.1 Edición básica ... 56
2.4.2 Tablas en la vista Authentic ... 61
- Tablas SPS ... 62
- Tablas CALS/HTML ... 63
- Iconos de edición para tablas CALS/HTML ... 67

2.4.3 Editar una BD ... 69
- Navegar por una tabla de BD ... 70
- Consultas de BD ... 71
- Modificar una tabla de BD ... 76

2.4.4 Trabajar con fechas ... 78
- Selector de fecha ... 79
- Entrada de texto ... 80

2.4.5 Definir entidades ... 81
2.4.6 Imágenes en la vista Authentic ... 83
2.4.7 Teclas de acceso rápido en la vista Authentic ... 84
2.5 Vista Explorador ... 85
2.6 Recursos globales de Altova ... 86
2.6.1 Definir recursos globales ... 87
- Archivos ... 90
- Carpetas ... 96
- Bases de datos ... 98

2.6.2 Usar recursos globales ... 100
- Asignar archivos y carpetas ... 101
- Cambiar de configuración ... 104

2.7 Control de código fuente ... 105
2.7.1 Configurar el control de código fuente ... 107
2.7.2 Sistemas de control de código fuente compatibles ... 108
2.7.3 Carpeta de trabajo local ... 110
2.7.4 Proyecto de la aplicación ... 111
2.7.5 Agregar al control de código fuente ... 113
2.7.6 Trabajar con control de código fuente ... 116
- Agregar o quitar del control de código fuente ... 117
- Proteger, desproteger ... 118
- Obtener archivos como archivos de solo lectura ... 121
- Copiar y compartir desde el control de código fuente ... 123
- Cambiar control de código fuente ... 126

2.8 Authentic Desktop en Visual Studio ... 127
2.8.1 Instalar el complemento de Authentic Desktop para Visual Studio ... 128
2.8.2 Diferencias entre la versión independiente y la versión para Visual Studio ... 129
2.9 Authentic Desktop en Eclipse ... 130
2.9.1 Instalar el complemento de Authentic Desktop para Eclipse ... 131
2.9.2 Puntos de entrada de Authentic Desktop en Eclipse ... 137
2.10 Comandos de menú ... 140
2.10.1 Menú Archivo ... 141
- Nuevo ... 142
- Abrir ... 144
- Volver a cargar ... 150

- Codificación ... 151
- Cerrar, Cerrar todos, Cerrar documentos inactivos ... 152
- Guardar, Guardar como, Guardar todos ... 153
- Enviar por correo electrónico ... 159
- Imprimir ... 161
- Vista previa de impresión, Configurar impresión ... 162
- Archivos recientes, Salir ... 163

2.10.2 Menú Edición ... 164

- Deshacer, Rehacer ... 165
- Cortar, Copiar, Pegar, Eliminar ... 166
- Seleccionar todo ... 167
- Buscar, Buscar siguiente ... 168
- Reemplazar ... 169

2.10.3 Menú Proyecto ... 170

- Proyecto nuevo ... 173
- Abrir proyecto ... 174
- Volver a cargar el proyecto ... 175
- Cerrar el proyecto ... 176
- Guardar el proyecto, Guardar el proyecto como ... 177
- Control de código fuente ... 178
  - Abrir desde el control de código fuente ... 178
  - Habilitar control de código fuente ... 179
  - Obtener la versión más reciente ... 180
  - Obtener, Obtener carpetas ... 180
  - Desproteger, Proteger ... 182
  - Anular desprotección ... 184
  - Agregar al control de código fuente ... 184
  - Quitar del control de código fuente ... 185
  - Compartir desde el control de código fuente ... 186
  - Mostrar historial ... 188
  - Mostrar diferencias ... 189
  - Mostrar propiedades ... 190
  - Actualizar estado ... 191
  - Administrador del control de código fuente ... 191
  - Cambiar control de código fuente ... 191
- Agregar archivos al proyecto ... 193
- Agregar recurso global al proyecto ... 194
- Agregar URL al proyecto ... 195
- Agregar archivo activo al proyecto ... 196
- Agregar archivo activo y relacionados al proyecto ... 197
- Agregar carpeta de proyecto al proyecto ... 198
- Agregar carpeta externa al proyecto ... 199
- Agregar carpeta web externa al proyecto ... 203
- Propiedades ... 208
- Proyectos usados recientemente ... 211

2.10.4 Menú XML ... 212
- Comprobar formato XML ... 213
- Validar el documento XML ... 215
- Validar el documento XML en el servidor ... 217

2.10.5 Menú XSL/XQuery ... 218
- Transformación XSL ... 219
- Transformación XSL-FO ... 220
- Parámetros de XSL / Variables de XQuery ... 222

2.10.6 Menú Authentic ... 226
- Documento nuevo ... 227
- Editar datos de una base de datos ... 228
- Editar una hoja de estilos de StyleVision ... 229
- Seleccionar y editar una fila nueva con datos XML ... 230
- Definir entidades XML ... 231
- Ver marcado ... 233
- Anexar/insertar/duplicar/eliminar fila ... 234
- Subir/bajar fila ... 235
- Generar documento HTML, RTF, PDF o Word 2007+ ... 236

2.10.7 Menú Explorador ... 237
- Atrás ... 238
- Adelante ... 239
- Detener ... 240
- Actualizar ... 241
- Fuentes ... 242
- Ventana separada ... 243

2.10.8 Menú Herramientas ... 244
- Ortografía ... 245
- Opciones de ortografía ... 249
- Recursos globales ... 253
- Configuración activa ... 254
- Personalizar ... 255
  - Comandos ... 255
  - Barras de herramientas ... 257
  - Teclado ... 258
  - Menú ... 265
  - Complementos ... 267
  - Opciones ... 269
  - Menú contextual Personalizar ... 269
- Restaurar barras de herramientas y ventanas ... 272
- Opciones ... 273
  - Archivo ... 273
  - Tipos de archivo ... 275
  - Vista ... 276
  - Codificación ... 277
  - XSL ... 277

Control de código fuente ... 280
2.10.9 Menú Ventanas ... 281
- En cascada ... 282
- En mosaico horizontal ... 283
- En mosaico vertical ... 284
- Ventana de proyecto ... 285
- Ventana de información ... 286
- Ayudantes de entrada ... 287
- Ventanas de resultados ... 288
- Proyecto y ayudantes de entrada ... 289
- Activar o desactivar todas ... 290
- Lista de ventanas abiertas ... 291
2.10.10 Menú Ayuda ... 292
- Contenido, Índice, Buscar ... 293
- Mapa de teclado ... 294
- Activación, formulario de pedido, registro, actualizaciones ... 295
- Otros comandos ... 297
2.10.11 Línea de comandos ... 298

3 Referencia del programador 300

3.1 ...Editor de scripts ... 302
3.1.1 Información general ... 304
- Proyectos de scripting en Authentic Desktop ... 305
- La interfaz gráfica del editor de scripts ... 307
- Componentes de un proyecto de scripts ... 312
3.1.2 Crear un proyecto de scripting ... 314
3.1.3 Declaraciones globales ... 316
3.1.4 Formularios ... 318
- Crear un formulario nuevo ... 319
- Diseño de formularios y objetos ... 321
- Eventos de formularios ... 324
3.1.5 Eventos ... 326
3.1.6 Macros ... 330
- Crear y editar una macro ... 331
- Ejecutar una macro ... 333
- Depurar una macro ... 336
3.1.7 Notas sobre programación ... 337
- Comandos integrados ... 339
Uso de los formularios y comandos ... 346
3.1.8 Migración al editor de scripts versión 2010 y superior ... 348
3.2 ...Complementos de Authentic Desktop para entornos IDE ... 351
3.2.1 Cómo registrar complementos para entornos IDE ... 352

3.2.2 Controles ActiveX ... 353
3.2.3 Configuración XML ... 354
3.2.4 Archivos de muestra ATL ... 358
- Descripción de interfaz (IDL) ... 359
- Definición de clase ... 361
- Implementación ... 362
3.2.5 IXMLSpyPlugIn ... 365
- OnCommand ... 366
- OnUpdateCommand ... 367
- OnEvent ... 368
- GetUIModifications ... 371
- GetDescription ... 372
3.3 ...Application API ... 373
3.3.1 Overview ... 375
- Object Model ... 376
- Programming Languages ... 377
JScript ... 378
Start Application ... 378
Simple Document Access ... 379
Iteration ... 381
Error Handling ... 383
Events ... 384
Example: Bubble Sort Dynamic Tables ... 385
VBScript ... 387
Events ... 387
Example: Using Events ... 388
C# ... 389
Add Reference to Authentic Desktop API ... 395
Application Startup and Shutdown ... 395
Opening Documents ... 396
Iterating through Open Documents ... 397
Errors and COM Output Parameters ... 398
Events ... 399
Java ... 400
Example Java Project ... 402
Application Startup and Shutdown ... 405
Simple Document Access ... 406
Iterations ... 407
Use of Out-Parameters ... 407
Event Handlers ... 408
3.3.2 Interfaces ... 409
- Application ... 410
Events ... 411
OnBeforeOpenDocument ... 411
OnBeforeOpenProject ... 412
OnDocumentOpened ... 412
OnProjectOpened ... 413
ActiveDocument ... 413

AddMacroMenuItem ... 413
AddXSLT_XQParameter ... 414
Application ... 414
ClearMacroMenu ... 414
CreateXMLSchemaFromDBStructure ... 415
CurrentProject ... 415
Dialogs ... 415
Documents ... 416
Edition ... 416
FindInFiles ... 416
GetDatabaseImportElementList ... 416
GetDatabaseSettings ... 417
GetDatabaseTables ... 417
GetExportSettings ... 418
GetTextImportElementList ... 418
GetTextImportExportSettings ... 419
GetXSLT_XQParameterCount ... 420
GetXSLT_XQParameterName ... 420
GetXSLT_XQParameterXPath ... 420
ImportFromDatabase ... 420
ImportFromSchema ... 421
ImportFromText ... 422
ImportFromWord ... 423
IsAPISupported ... 423
MajorVersion ... 423
MinorVersion ... 424
NewProject ... 424
OpenProject ... 424
Parent ... 425
Quit ... 425
ReloadSettings ... 425
RemoveXSLT_XQParameter ... 426
RunMacro ... 426
ScriptingEnvironment ... 426
ServicePackVersion ... 427
ShowApplication ... 427
ShowFindInFiles ... 427
ShowForm ... 427
Status ... 428
URLDelete ... 428
URLMakeDirectory ... 428
Visible ... 429
WarningNumber ... 429
WarningText ... 429
– AuthenticContextMenu ... 430

CountItems ... 430
DeleteItem ... 430
GetItemText ... 430
InsertItem ... 430
SetItemText ... 431
– AuthenticDataTransfer ... 432
dropEffect ... 432
getData ... 432
ownDrag ... 433
type ... 433
– AuthenticEventContext ... 434
EvaluateXPath ... 434
GetEventContextType ... 434
GetNormalizedTextValue ... 434
GetVariableValue ... 435
GetXMLNode ... 435
IsAvailable ... 435
SetVariableValue ... 436
– AuthenticRange ... 437
AppendRow ... 438
Application ... 439
CanPerformAction ... 439
CanPerformActionWith ... 439
Clone ... 440
CollapsToBegin ... 440
CollapsToEnd ... 440
Copy ... 441
Cut ... 441
Delete ... 441
DeleteRow ... 442
DuplicateRow ... 442
EvaluateXPath ... 443
ExpandTo ... 443
FirstTextPosition ... 443
FirstXMLData ... 444
FirstXMLDataOffset ... 445
GetElementAttributeNames ... 446
GetElementAttributeValue ... 447
GetElementHierarchy ... 447
GetEntityNames ... 447
GetVariableValue ... 448
Goto ... 448
GotoNext ... 448
GotoNextCursorPosition ... 449
GotoPrevious ... 450

GotoPreviousCursorPosition ... 450
HasElementAttribute ... 451
InsertEntity ... 451
InsertRow ... 452
IsCopyEnabled ... 452
IsCutEnabled ... 452
IsDeleteEnabled ... 453
IsEmpty ... 453
IsEqual ... 453
IsFirstRow ... 453
IsInDynamicTable ... 454
IsLastRow ... 454
IsPasteEnabled ... 454
IsSelected ... 455
IsTextStateApplied ... 455
LastTextPosition ... 455
LastXMLData ... 456
LastXMLDataOffset ... 457
MoveBegin ... 458
MoveEnd ... 458
MoveRowDown ... 459
MoveRowUp ... 459
Parent ... 459
Paste ... 460
PerformAction ... 460
Select ... 461
SelectNext ... 461
SelectPrevious ... 462
SetElementAttributeValue ... 463
SetFromRange ... 464
SetVariableValue ... 464
Text ... 464
− AuthenticView ... 466
Events ... 466
OnBeforeCopy ... 466
OnBeforeCut ... 467
OnBeforeDelete ... 467
OnBeforeDrop ... 468
OnBeforePaste ... 468
OnBeforeSave ... 469
OnDragOver ... 469
OnKeyboardEvent ... 470
OnLoad ... 470
OnMouseEvent ... 471
OnSelectionChanged ... 471
OnToolbarButtonClicked ... 472
OnToolbarButtonExecuted ... 473

OnUserAddedXMLNode ... 473
Application ... 474
AsXMLString ... 474
ContextMenu ... 474
CreateXMLNode ... 475
DisableAttributeEntryHelper ... 475
DisableElementEntryHelper ... 475
DisableEntityEntryHelper ... 475
DocumentBegin ... 476
DocumentEnd ... 476
DoNotPerformStandardAction ... 476
EvaluateXPath ... 476
Event ... 477
EventContext ... 477
GetToolbarButtonState ... 477
Goto ... 478
IsRedoEnabled ... 478
IsUndoEnabled ... 479
MarkupVisibility ... 479
Parent ... 479
Print ... 479
Redo ... 480
Selection ... 480
SetToolbarButtonState ... 481
Undo ... 481
UpdateXMLInstanceEntities ... 481
WholeDocument ... 482
XMLDataRoot ... 482
− CodeGeneratorDlg ... 483
Application ... 483
CPPSettings_DOMType ... 484
CPPSettings_GenerateVC6ProjectFile ... 484
CPPSettings_GenerateGCCMakefile ... 484
CPPSettings_GenerateVSProjectFile ... 485
CPPSettings_LibraryType ... 485
CPPSettings_UseMFC ... 485
CSharpSettings_ProjectType ... 485
OutputPath ... 486
OutputPathDialogAction ... 486
OutputResultDialogAction ... 486
Parent ... 487
ProgrammingLanguage ... 487
PropertySheetDialogAction ... 487
TemplateFileName ... 488
− DatabaseConnection ... 489

ADOConnection ... 489
AsAttributes ... 490
CommentIncluded ... 490
CreateMissingTables ... 490
CreateNew ... 491
DatabaseKind ... 491
DatabaseSchema ... 491
ExcludeKeys ... 491
File ... 492
ForeignKeys ... 492
ImportColumnsType ... 492
IncludeEmptyElements ... 492
NullReplacement ... 493
NumberDateTimeFormat ... 493
ODBCConnection ... 493
PrimaryKeys ... 493
SchemaExtensionType ... 494
SchemaFormat ... 494
SQLSelect ... 494
TextFieldLen ... 494
UniqueKeys ... 495
– Dialogs ... 496
Application ... 496
CodeGeneratorDlg ... 496
FileSelectionDlg ... 497
Parent ... 497
SchemaDocumentationDlg ... 497
GenerateSampleXMLDlg ... 497
DTDSchemaGeneratorDlg ... 498
FindInFilesDlg ... 498
– Document ... 499
Events ... 500
OnBeforeSaveDocument ... 500
OnBeforeCloseDocument ... 501
OnBeforeValidate ... 502
OnCloseDocument ... 502
OnViewActivation ... 503
Application ... 503
AssignDTD ... 504
AssignSchema ... 504
AssignXSL ... 504
AssignXSLFO ... 505
AsXMLString ... 505
AuthenticView ... 505
Close ... 506
ConvertDTDOrSchema ... 506

ConvertDTDOrSchemaEx ... 507
ConvertToWSDL20 ... 507
CreateChild ... 508
CreateDBStructureFromXMLSchema ... 508
CreateSchemaDiagram ... 509
CurrentViewMode ... 509
DataRoot ... 509
DocEditView ... 510
Encoding ... 510
EndChanges ... 510
ExecuteXQuery ... 511
ExportToDatabase ... 511
ExportToText ... 512
FlattenDTDOrSchema ... 513
FullName ... 514
GenerateDTDOrSchema ... 514
GenerateDTDOrSchemaEx ... 515
GenerateProgramCode ... 515
GenerateSampleXML ... 515
GenerateSchemaDocumentation ... 515
GetDBStructureList ... 516
GetExportElementList ... 516
GetPathName (obsolete) ... 517
GridView ... 517
IsModified ... 517
IsValid ... 518
IsValidEx ... 518
IsWellFormed ... 519
Name ... 519
Parent ... 519
Path ... 520
RootElement ... 520
Save ... 520
SaveAs ... 521
Saved ... 521
SaveInString ... 521
SaveToURL ... 522
SetActiveDocument ... 522
SetEncoding (obsolete) ... 522
SetExternalIsValid ... 524
SetPathName (obsolete) ... 524
StartChanges ... 524
Suggestions ... 525
SwitchViewMode ... 525
TextView ... 525

Title ... 525
TransformXSL ... 526
TransformXSLEx ... 526
TransformXSLFO ... 526
UpdateViews ... 527
UpdateXMLData ... 527
− Documents ... 528
Count ... 528
Item ... 529
NewAuthenticFile ... 529
NewFile ... 529
NewFileFromText ... 530
OpenAuthenticFile ... 530
OpenFile ... 530
OpenURL ... 531
OpenURLDialog ... 532
− DTDSchemaGeneratorDlg ... 533
Application ... 533
AttributeTypeDefinition ... 533
DTDSchemaFormat ... 534
FrequentElements ... 534
GlobalAttributes ... 534
MaxEnumLength ... 534
MergeAllEqualNamed ... 535
OnlyStringEnums ... 535
OutputPath ... 535
OutputPathDialogAction ... 535
Parent ... 536
ResolveEntities ... 536
TypeDetection ... 536
ValueList ... 536
− ElementList ... 537
Count ... 537
Item ... 537
RemoveElement ... 537
− ElementListItem ... 539
ElementKind ... 539
FieldCount ... 539
Name ... 539
RecordCount ... 539
− ExportSettings ... 541
CreateKeys ... 541
ElementList ... 541
EntitiesToText ... 541
ExportAllElements ... 542

ExportCompleteXML ... 542
FromAttributes ... 542
FromSingleSubElements ... 542
FromTextValues ... 542
IndependentPrimaryKey ... 543
Namespace ... 543
StartFromElement ... 543
SubLevelLimit ... 543
– FileSelectionDlg ... 544
Application ... 544
DialogAction ... 544
FullName ... 545
Parent ... 545
– FindInFilesDlg ... 546
AdvancedXMLSearch ... 546
Application ... 547
DoReplace ... 547
FileExtension ... 547
Find ... 547
IncludeSubfolders ... 547
MatchCase ... 548
MatchWholeWord ... 548
Parent ... 548
RegularExpression ... 548
Replace ... 549
ReplaceOnDisk ... 549
SearchInProjectFilesDoExternal ... 549
SearchLocation ... 549
ShowResult ... 549
StartFolder ... 550
XMLAttributeContents ... 550
XMLAttributeNames ... 550
XMLCData ... 550
XMLComments ... 550
XMLElementContents ... 551
XMLElementNames ... 551
XMLPI ... 551
XMLRest ... 551
– FindInFilesResult ... 552
Application ... 552
Count ... 552
Document ... 552
Item ... 553
Parent ... 553
Path ... 553

– FindInFilesResultMatch ... 554
Application ... 554
Length ... 554
Line ... 554
LineText ... 555
Parent ... 555
Position ... 555
Replaced ... 555
– FindInFilesResults ... 556
Application ... 556
Count ... 556
Item ... 556
Parent ... 556
– GenerateSampleXMLDlg ... 557
Application ... 557
ChoiceMode ... 557
ConsiderSampleValueHints ... 558
ContentOfNillableElementsIsNonMandatory ... 558
FillAttributesWithSampleData ... 558
FillElementsWithSampleData ... 558
FillWithSampleData - obsolete ... 559
LocalNameOfRootElement ... 559
NamespaceURIOfRootElement ... 559
NonMandatoryAttributes ... 559
NonMandatoryElements ... 559
Optimization - obsolete ... 560
OptionsDialogAction ... 560
Parent ... 560
RepeatCount ... 560
SampleValueHints ... 561
SchemaOrDTDAssignment ... 561
TakeFirstChoice - obsolete ... 561
TryToUseNonAbstractTypes ... 561
– GridView ... 562
Events ... 562
OnBeforeDrag ... 562
OnBeforeDrop ... 562
OnBeforeStartEditing ... 563
OnEditingFinished ... 563
OnFocusChanged ... 564
CurrentFocus ... 564
Deselect ... 565
IsVisible ... 565
Select ... 565
SetFocus ... 565
– SchemaDocumentationDlg ... 566

AllDetails ... 567
Application ... 567
CreateDiagramsFolder ... 567
DiagramFormat ... 568
EmbedCSSInHTML ... 568
EmbedDiagrams ... 568
GenerateRelativeLinks ... 569
IncludeAll ... 569
IncludeAttributeGroups ... 569
IncludeComplexTypes ... 569
IncludeGlobalAttributes ... 570
IncludeGlobalElements ... 570
IncludeGroups ... 570
IncludeIndex ... 571
IncludeLocalAttributes ... 571
IncludeLocalElements ... 571
IncludeRedefines ... 571
IncludeReferencedSchemas ... 572
IncludeSimpleTypes ... 572
MultipleOutputFiles ... 572
OptionsDialogAction ... 573
OutputFile ... 573
OutputFileDialogAction ... 573
OutputFormat ... 574
Parent ... 574
ShowAnnotations ... 574
ShowAttributes ... 574
ShowChildren ... 575
ShowDiagram ... 575
ShowEnumerations ... 575
ShowIdentityConstraints ... 576
ShowNamespace ... 576
ShowPatterns ... 576
ShowProgressBar ... 577
ShowProperties ... 577
ShowResult ... 577
ShowSingleFacets ... 578
ShowSourceCode ... 578
ShowType ... 578
ShowUsedBy ... 578
SPSFile ... 579
UseFixedDesign ... 579
− SpyProject ... 580
CloseProject ... 580
ProjectFile ... 580

RootItems ... 580
SaveProject ... 581
SaveProjectAs ... 581
− SpyProjectItem ... 582
ChildItems ... 582
FileExtensions ... 582
ItemType ... 582
Name ... 583
Open ... 583
ParentItem ... 583
Path ... 583
ValidateWith ... 583
XMLForXSLTransformation ... 584
XSLForXMLTransformation ... 584
XSLTransformationFileExtension ... 584
XSLTransformationFolder ... 584
− SpyProjectItems ... 585
AddFile ... 585
AddFolder ... 585
AddURL ... 585
Count ... 586
Item ... 586
RemoveItem ... 586
− TextImportExportSettings ... 587
CommentIncluded ... 587
DestinationFolder ... 587
EnclosingCharacter ... 587
Encoding ... 588
EncodingByteOrder ... 588
FieldDelimiter ... 588
FileExtension ... 588
HeaderRow ... 588
ImportFile ... 589
RemoveDelimiter ... 589
RemoveNewline ... 589
− TextView ... 590
Events ... 590
OnBeforeShowSuggestions ... 590
OnChar ... 591
Application ... 591
GetRangeText ... 592
GoToLineChar ... 592
Length ... 592
LineCount ... 592
LineFromPosition ... 592

LineLength ... 593
MoveCaret ... 593
Parent ... 593
PositionFromLine ... 593
ReplaceText ... 593
SelectionEnd ... 594
SelectionStart ... 594
SelectText ... 594
SelText ... 594
Text ... 595
– XMLData ... 596
AppendChild ... 597
CountChildren ... 597
CountChildrenKind ... 598
EraseAllChildren ... 598
EraseChild ... 598
EraseCurrentChild ... 599
GetChild ... 599
GetChildAttribute ... 600
GetChildElement ... 600
GetChildKind ... 600
GetCurrentChild ... 601
GetFirstChild ... 601
GetNamespacePrefixForURI ... 602
GetNextChild ... 602
GetTextValueXMLDecoded ... 603
HasChildren ... 603
HasChildrenKind ... 604
InsertChild ... 604
InsertChildAfter ... 604
InsertChildBefore ... 605
IsSameNode ... 605
Kind ... 605
MayHaveChildren ... 605
Name ... 606
Parent ... 606
SetTextValueXMLEncoded ... 606
TextValue ... 607
3.3.3 Enumerations ... 608
– ENUMApplicationStatus ... 609
– SPYAttributeTypeDefinition ... 610
– SPYAuthenticActions ... 611
– SPYAuthenticDocumentPosition ... 612
– SPYAuthenticElementActions ... 613
– SPYAuthenticElementKind ... 614

– SPYAuthenticMarkupVisibility ... 615
– SPYAuthenticToolbarButtonState ... 616
– SPYDatabaseKind ... 617
– SPYDialogAction ... 618
– SPYDOMType ... 619
– SPYDTDSchemaFormat ... 620
– SPYEncodingByteOrder ... 621
– SPYExportNamespace ... 622
– SPYFindInFilesSearchLocation ... 623
– SPYFrequentElements ... 624
– SPYImageKind ... 625
– SPYImportColumnsType ... 626
– SPYKeyEvent ... 627
– SPYKeyStatus ... 628
– SPYLibType ... 629
– SPYLoading ... 630
– SPYMouseEvent ... 631
– SPYNumberDateTimeFormat ... 632
– SPYProgrammingLanguage ... 633
– SPYProjectItemTypes ... 634
– SPYProjectType ... 635
– SpySampleXMLGenerationChoiceMode ... 636
– SPYSampleXMLGenerationOptimization (Obsolete) ... 637
– SpySampleXMLGenerationSampleValueHints ... 638
– SPYSampleXMLGenerationSchemaOrDTDAssignment ... 639
– SPYSchemaDefKind ... 640
– SPYSchemaDocumentationFormat ... 641
– SPYSchemaExtensionType ... 642
– SPYSchemaFormat ... 643
– SPYTextDelimiters ... 644
– SPYTextEnclosing ... 645
– SPYTypeDetection ... 646
– SPYURLTypes ... 647
– SPYValidateXSDVersion ... 648
– SPYValidateErrorFormat ... 649
– SPYViewModes ... 650
– SPYVirtualKeyMask ... 651
– SPYXMLDataKind ... 652

3.4 ...ActiveX Integration ... 653

3.4.1 Integration at Application Level ... 654

3.4.2 Integration at Document Level ... 655

– Use AuthenticDesktopControl ... 656
– Use AuthenticDesktopControlDocument ... 657
– Use AuthenticDesktopControlPlaceHolder ... 658
– Query Authentic Desktop Commands ... 659

3.4.3 Programming Languages 660
- C# 661
Introduction 661
Placing the AuthenticDesktopControl 661
- HTML 662
Integration at Application Level 662
Instantiate the Control 662
Add Button to Open Default Document 662
Connect to Custom Events 662
Integration at Document Level 663
Instantiate the AuthenticDesktopControl 663
Create Editor Window 663
Create Project Window 664
Create Placeholder for Helper Windows 664
- Java 665
Example Java Project 666
Creating the ActiveX Controls 674
Loading Data in the Controls 675
Basic Event Handling 675
Menus 676
UI Update Event Handling 677
Creating an XML Tree 678
3.4.4 Command Table for Authentic Desktop 681
- File Menu 682
- Edit Menu 683
- Project Menu 684
- XML Menu 686
- XSL/XQuery Menu 687
- Authentic Menu 688
- View Menu 689
- Browser Menu 690
- Tools Menu 691
- Window Menu 692
- Help Menu 693
- Misc Menu 694
3.4.5 Accessing AuthenticDesktopAPI 701
3.4.6 Object Reference 702
- Authentic DesktopCommand 703
Accelerator 703
ID 703
IsSeparator 703
Label 704
StatusText 704
SubCommands 704
ToolTip 704
- Authentic DesktopCommands 705

Count ... 705
Item ... 705
− AuthenticDesktopControl ... 706
Properties ... 706
Appearance ... 706
Application ... 707
BorderStyle ... 707
CommandsList ... 707
EnableUserPrompts ... 707
IntegrationLevel ... 707
MainMenu ... 708
Toolbars ... 708
Methods ... 708
Exec ... 708
Open ... 709
QueryStatus ... 709
Events ... 709
OnCloseEditingWindow ... 710
OnContextChanged ... 710
OnDocumentOpened ... 710
OnFileChangedAlert ... 710
OnLicenseProblem ... 710
OnOpenedOrFocused ... 711
OnToolWindowUpdated ... 711
OnUpdateCmdUI ... 711
OnValidationWindowUpdated ... 711
− AuthenticDesktopControlDocument ... 713
Properties ... 713
Appearance ... 713
BorderStyle ... 714
Document ... 714
IsModified ... 714
Path ... 714
ReadOnly ... 714
Methods ... 715
Exec ... 715
New ... 715
Open ... 715
QueryStatus ... 716
Reload ... 716
Save ... 716
SaveAs ... 716
Events ... 717
OnActivate ... 717
OnContextChanged ... 717
OnDocumentClosed ... 717
OnDocumentOpened ... 717
OnDocumentSaveAs ... 718
OnFileChangedAlert ... 718
OnModifiedFlagChanged ... 718
OnSetEditorTitle ... 718
− AuthenticDesktopControlPlaceHolder ... 719

Properties ... 719
Label ... 719
PlaceholderWindowID ... 719
Project ... 720
Methods ... 720
OpenProject ... 720
CloseProject ... 720
Events ... 720
OnModifiedFlagChanged ... 721
OnSetLabel ... 721
− Enumerations ... 722
ICActiveXIntegrationLevel ... 722
AuthenticDesktopControlPlaceholderWindow ... 722

4 Anexos 724

4.1 ...Datos técnicos ... 725
4.1.1 Requisitos de SO y memoria ... 726
4.1.2 Validador XML de Altova ... 727
4.1.3 Motores XSLT y XQuery de Altova ... 728
4.1.4 Compatibilidad con Unicode ... 729
4.1.5 Uso de Internet ... 730
4.2 ...Información sobre licencias ... 731
4.2.1 Distribución electrónica de software ... 732
4.2.2 Derechos de propiedad intelectual ... 733
4.2.3 Altova Authentic: contrato de licencia para el usuario final ... 734

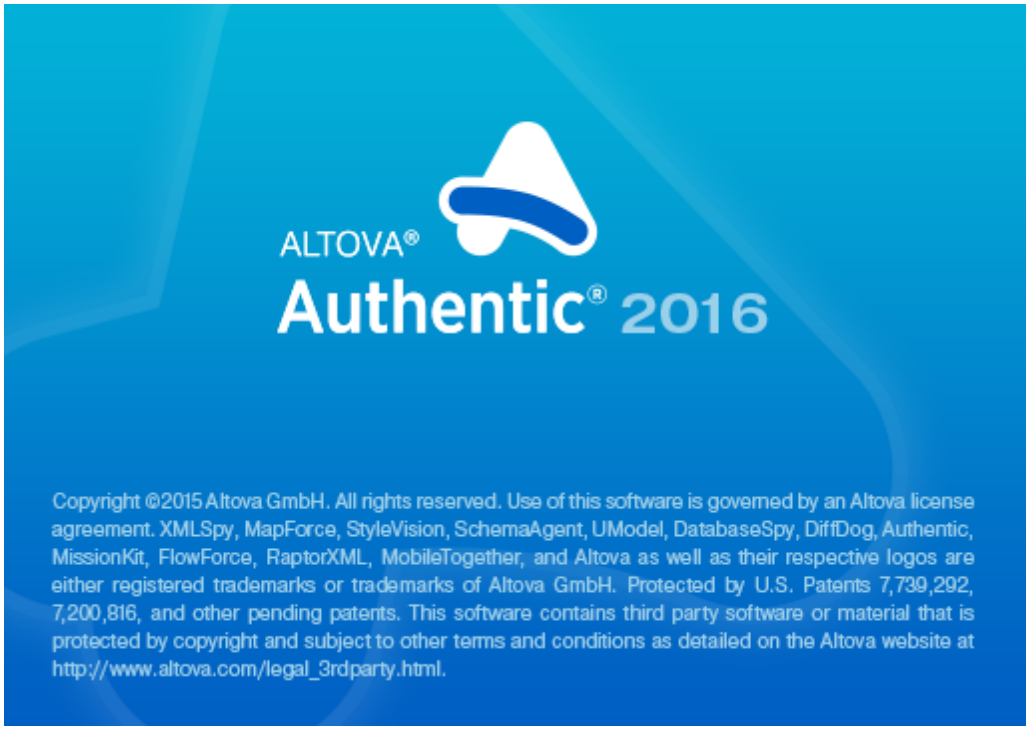
Índice

Altova Authentic 2016 Community Edition

Altova Authentic 2016 Community Edition

Altova Authentic 2016 Community Edition

Altova Authentic 2016 Community Edition es una innovadora herramienta visual para publicar documentos XML sin necesidad de preocuparse por los aspectos técnicos del lenguaje XML. Authentic Desktop es compatible con Windows 10, Windows 8, Windows 7, Windows Vista, Windows XP y Windows Server 2003/2008/2012. La edición Enterprise Edition se puede usar en equipos de 64 y 32 bits.


ALTOVA®
Authentic® 2016
Copyright ©2015 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, MobileTogether, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 7,739,292, 7,200,816, and other pending patents. This software contains third party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at http://www.altova.com/legal_3rdparty.html.

Última actualización: 24-09-2015

© 2015 Altova GmbH

Altova Authentic 2016 Community Edition

Guía del usuario y de referencia

Guía del usuario y de referencia

Esta guía del usuario y de referencia contiene un tutorial y un manual que explica las características básicas de la vista Authentic para que pueda empezar a trabajar con la aplicación. También incluye una extensa referencia que describe todas las características de la interfaz.

La presente documentación está dividida en estas secciones:

- La introducción describe la interfaz gráfica del usuario y el entorno de Authentic Desktop.
- El tutorial le ayudará a empezar a trabajar con Authentic Desktop.
- La sección sobre la vista Authentic describe esta vista WYSIWYG para trabajar con documentos XML. La vista Authentic permite al usuario escribir y editar documentos XML como si fueran sencillos documentos de texto o formularios interactivos. El marcado XML se oculta para que el usuario pueda concentrarse en el contenido del documento. La vista Authentic es la vista principal de Authentic Desktop.
- La sección Vista Explorador describa la vista en la que puede transformar documentos XML instantáneamente y verlos en una ventana del explorador web.
- La sección Recursos globales describe esta característica de los productos de Altova que permite cambiar de recursos de forma rápida y sencilla.
- Dos apartados sobre cómo integrar Authentic Desktop en Visual Studio y Eclipse.
- La sección Comandos de menú describe todas las ventanas y comandos de menú de Authentic Desktop.

Rutas de acceso de archivos en Windows XP, Windows Vista, Windows 7 y Windows 8

Las rutas de acceso de archivo utilizadas en esta documentación no son iguales para todos los sistemas operativos.

- *Carpeta (Mis) Documentos:* esta carpeta se encuentra por defecto en la ubicación que aparece a continuación. Los archivos de ejemplo se encuentran en una subcarpeta de esta carpeta.

Windows XP	`C:/Documents and Settings/<usuario>/Mis Documentos`
Windows Vista, Windows 7/8	`C:/Users/<usuario>/Documentos`

- *Carpeta de la aplicación:* esta es la carpeta en la que se encuentra su aplicación de Altova. Esta es la ruta de acceso predeterminada de la carpeta de la aplicación:

Windows XP	`C:/Archivos de programa/Altova`
Windows Vista, Windows 7, Windows 8	`C:/Archivos de programa/Altova`
Versión de 32 bits en sistemas operativos de 64 bits	`C:/Archivos de programa (x86)/Altova`

Nota: Authentic Desktop también es compatible con Windows Server 2003, Windows 2008 y Windows Server 2012.

© 2015 Altova GmbH

1 Interfaz y entorno

Esta introducción describe:

- la interfaz gráfica del usuario (IGU) y
- el entorno de aplicación.

La sección dedicada a la IGU empieza con una introducción sobre la IGU y después describe en detalle las diferentes ventanas de la interfaz. También explica como cambiar el tamaño de las ventanas, moverlas y cómo trabajar con ellas en general.

La sección dedicada al entorno de aplicación describe las opciones de configuración con las que puede controlar cómo se visualizan y editan los diferentes tipos de archivo. También explica cómo se puede personalizar la aplicación. Además aprenderá dónde se guardan los archivos de ejemplo importantes y el tutorial. Al final de la sección, encontrará enlaces al sitio web de Altova, donde podrá consultar una lista completa con todas las funciones y características de la aplicación. En el sitio web también encontrará el *Manual del usuario* en diferentes formatos, información sobre las opciones de soporte técnico disponibles y detalles sobre los demás productos de Altova.

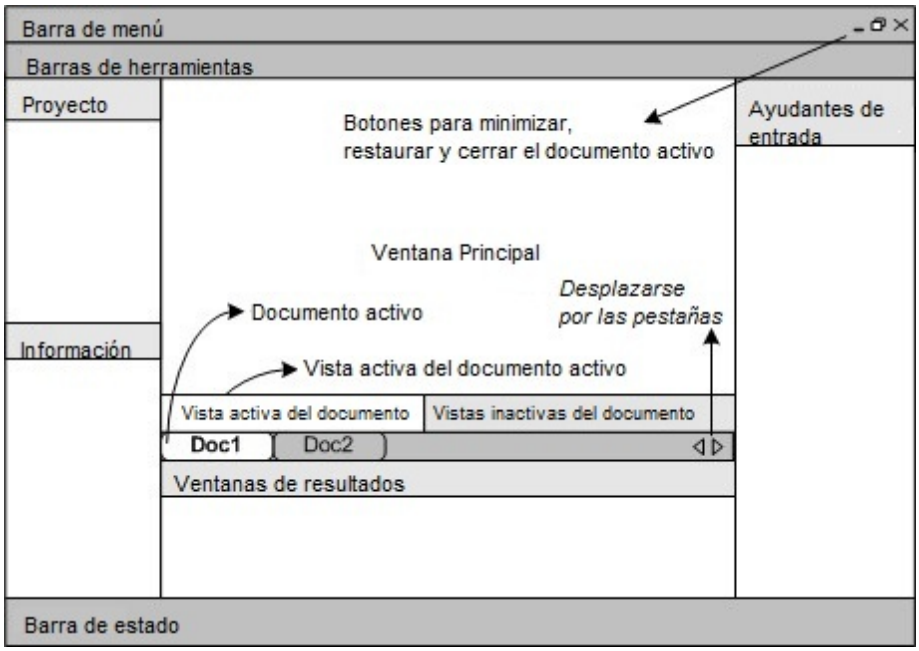
© 2015 Altova GmbH

1.1 La interfaz gráfica de usuario (IGU)

La interfaz gráfica de usuario (IGU) incluye una ventana principal y varias barras laterales (*imagen siguiente*). Las barras laterales están situadas por defecto alrededor de la ventana principal y están ordenadas en varios grupos:

- Ventana Proyecto
- Ventana Información
- Ayudantes de entrada: Elementos, Atributos, Entidades, etc. (dependiendo del tipo de documento que esté activo)
- Ventanas de resultados: Mensajes

Los diferentes apartados de esta sección describen en detalle la ventana principal y las barras laterales.



La IGU también incluye una barra de menú, una barra de estado y barras de herramientas, todas ellas descritas en esta sección.

Activar o desactivar las barras laterales

Los grupos de barras laterales (ventana Proyecto, ventana Información, ayudantes de entrada y ventanas de resultados) se pueden mostrar u ocultar con solo pulsar los comandos correspondientes del menú **Ventanas**. Además puede ocultar las barras laterales (o un grupo de barras laterales organizadas en pestañas) si hace clic con el botón secundario sobre el título de la barra y selecciona el comando **Ocultar**.

Desacoplar y acoplar las barras laterales

Las ventanas de las barras laterales pueden desacoplarse de la IGU o acoplarse en la IGU. Cuando se acopla una ventana flotante, se acopla en la última posición donde estuvo acoplada. También puede acoplar una ventana dentro de otra ventana en forma de pestaña.

Hay varios métodos para acoplar o desacoplar una ventana:

- Haga clic con el botón secundario en la barra de título de una ventana y seleccione el comando pertinente (**Flotante** o **Acoplada**).
- Haga doble clic en la barra de título de la ventana. Si la ventana está acoplada, se desacoplará. Si está desacoplada, la ventana se acoplará en la última posición donde estuvo acoplada.
- Arrastre la ventana (usando su barra de título como controlador) y desacóplela de su posición. Arrastre una ventana flotante (por su barra de título) hasta la posición donde desee acoplarla. Aparecerán dos grupos de flechas azules. El grupo de flechas exterior permite acoplar la ventana en relación a la ventana de la aplicación (en relación al borde superior, inferior, izquierdo o derecho de la IGU). El grupo de flechas interior permite acoplar la ventana en relación a la ventana sobre la que está el puntero del mouse. Si coloca la ventana sobre el botón central del conjunto de flechas interior (o sobre la barra de título de una ventana), la ventana se acopla dentro de la ventana en forma de pestaña.

Para desacoplar una pestaña de una ventana, haga clic con el botón secundario sobre la pestaña. Para arrastrar una pestaña fuera de su ventana, simplemente haga clic sobre la pestaña y arrástrela.

Ocultar automáticamente las barras laterales

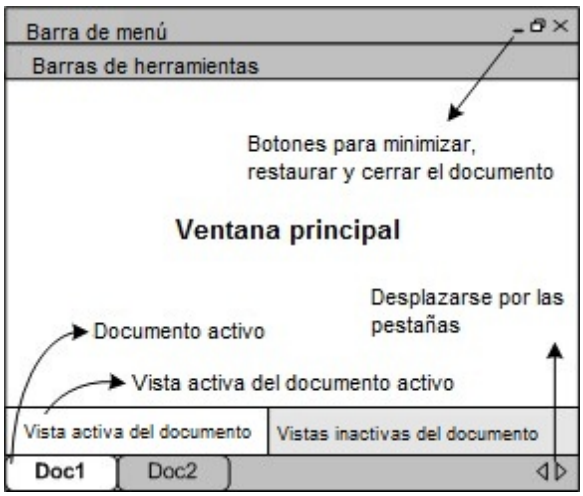
Las barras laterales acopladas se pueden minimizar en forma de botones que se sitúan en los bordes de la ventana de la aplicación. De esta manera tendrá más espacio para trabajar en la ventana principal y en las demás barras laterales. Al pasar el cursor sobre la barra minimizada, ésta se abrirá.

Para ocultar automáticamente las barras laterales o restaurarlas, haga clic en el icono en forma de chincheta que aparece en la barra de título de la barra lateral (o haga clic con el botón secundario y seleccione **Ocultar automáticamente**).

© 2015 Altova GmbH

1.1.1 Ventana principal

La ventana principal (*imagen siguiente*) es la ventana donde se visualizan y editan los documentos.



Archivos en la ventana principal

- Se pueden abrir y editar a la vez cualquier cantidad de archivos
- Cada documento que esté abierto tiene su propia ventana y pestaña (que tiene el nombre de archivo del documento) en la parte inferior de la ventana principal. Para activar un documento, haga clic en su pestaña.
- Si hay varios archivos abiertos, puede que algunas pestañas no sean visibles por falta de espacio en la barra de pestañas. Para ver todas las pestañas: (i) utilice los botones de desplazamiento situados a la derecha de la barra de pestañas o (ii) seleccione el documento pertinente en el menú **Ventanas**.
- Cuando se maximiza el documento activo, los botones **Minimizar**, **Restaurar** y **Cerrar** están situados en el lado derecho de la barra de menú. Cuando un documento está en cascada, en mosaico o minimizado, los botones **Maximizar**, **Restaurar** y **Cerrar** están situados en la barra de título de la ventana del documento.
- Cuando se maximiza un archivo, se maximizan también todos los archivos que estén abiertos.
- Puede organizar los archivos abiertos en cascada o en mosaico con los comandos del menú **Ventanas**.
- También puede activar los archivos abiertos en la misma secuencia en la que se abrieron si pulsa las teclas **Ctrl+Tabulador** o **Ctrl+F6**.
- Al hacer clic con el botón secundario en la pestaña de un documento aparece un menú contextual con varios comandos del menú **Archivo**, como **Imprimir** y **Cerrar**.

Vistas en la ventana principal

El documento activo se puede visualizar y editar en varias vistas. Las vistas disponibles aparecen en una barra situada sobre las pestañas de los documentos (*ver imagen anterior*) y la vista activa aparece resaltada. Para seleccionar una vista, haga clic en el botón de la vista o seleccione la vista pertinente en el menú **Vista**.

Las vistas disponibles pueden ser vistas de edición o vistas de explorador:

- **Vista Authentic:** para editar documentos XML basados en hojas de estilos StyleVision

Power Stylesheet en una interfaz gráfica.

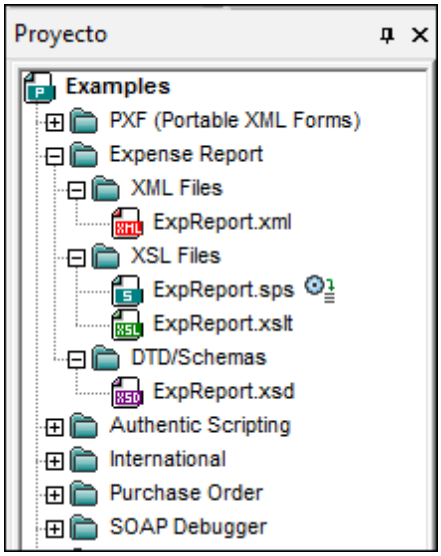
- **Vista Explorador:** una vista de explorador integrada compatible con CSS y hojas de estilos XSL.

Nota: La vista predeterminada para cada tipo de archivo se puede personalizar en el cuadro de diálogo **Herramientas | Opciones** (en el panel "Vista predeterminada" de la pestaña "Tipos de archivo").

© 2015 Altova GmbH

1.1.2 Ventana Proyecto

Un proyecto es un conjunto de archivos relacionados entre sí. Por ejemplo, en la imagen siguiente, el proyecto llamado `Examples` agrupa varios archivos de ejemplo en diferentes carpetas, cada una de ellas con sus propias subcarpetas. Por ejemplo, en el proyecto `Examples` la carpeta de ejemplo `Expense report` tiene subcarpetas para archivos XML, archivos XSL y archivos de esquema.



Por tanto, los proyectos son una manera de reunir archivos que se usan juntos y también de tener acceso a ellos más rápido. Además, puede definir archivos de esquema y XSLT para cada carpeta para procesar los archivos de cada carpeta por lotes.

Operaciones de proyecto

Los comandos para las operaciones de carpeta están en el menú **Proyecto** y algunos están también disponibles en los menús contextuales del proyecto y sus carpetas (al hacer clic con el botón secundario).

- En la ventana Proyecto se puede abrir sólo un proyecto. Cuando se crea un proyecto nuevo o se abre un proyecto ya existente, el proyecto de la ventana Proyecto se sustituye con el proyecto nuevo.
- Después de realizar cambios en un proyecto, es necesario guardar los cambios (clic en el comando **Proyecto | Guardar proyecto**). Si en un proyecto hay cambios sin guardar, junto al nombre del proyecto aparece un asterisco.
- El proyecto se representa en forma de árbol, con carpetas, archivos y otros recursos. Los recursos se pueden agregar en cualquier nivel y en cualquier subcarpeta.
- Las carpetas de proyecto son carpetas *semánticas* que representan una agrupación lógica de archivos. **No se corresponden** con la organización jerárquica de los archivos en su disco duro.
- Las carpetas pueden coincidir y estar relacionadas con directorios de su sistema de archivo. A esas carpetas las llamamos *carpetas externas* y en la ventana Proyecto se representan con un icono de carpeta amarilla (en contraste con las carpetas normales, que son verdes). Las carpetas externas de proyecto deben sincronizarse expresamente con el comando **Actualizar**.
- Cada carpeta puede contener varios tipos de archivo diferentes. También puede definir extensiones de archivo para cada carpeta (en el cuadro de diálogo "Propiedades" de la

carpeta) para guardar los archivos más comunes en varias carpetas. Cuando se añade un archivo a la carpeta primaria, el archivo se añade automáticamente a la subcarpeta a la que se asignaron archivos con ese tipo de extensión.

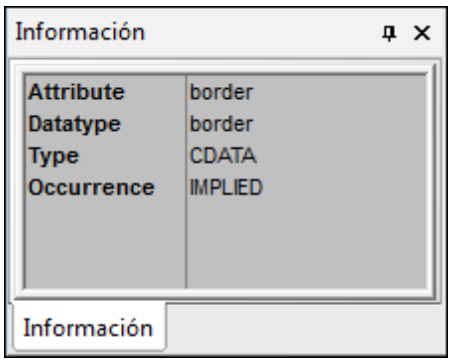
- En la ventana Proyecto puede arrastrar carpetas y colocarlas en otra carpeta o en otra posición de la misma carpeta. Los archivos, sin embargo, sólo se pueden arrastrar a otra carpeta, pero no se pueden mover a otra posición de la misma carpeta (en la que los archivos están organizados alfabéticamente). Además, puede arrastrar archivos y carpetas desde el explorador de archivos de Windows hasta la ventana Proyecto.
- Cada carpeta tiene un conjunto de propiedades que se definen en el cuadro de diálogo "Propiedades" de cada carpeta. Estas propiedades incluyen extensiones de archivos asignadas a la carpeta, el esquema con el que validar los archivos XML, el archivo XSLT con el que transformar los archivos XML, etc.
- Para procesar por lotes los archivos de una carpeta, haga clic con el botón secundario en la carpeta y seleccione el comando pertinente en el menú contextual (p. ej. **Validar el documento XML** o **Comprobar formato XML**).

Nota: Para habilitar o deshabilitar la ventana Proyecto, utilice el menú **Ventanas**.

© 2015 Altova GmbH

1.1.3 Ventana Información

La ventana Información (*imagen siguiente*) muestra información sobre el elemento o el atributo donde está situado el cursor.



Esta ventana se puede ocultar/mostrar desde el menú **Ventanas**.

1.1.4 Ayudantes de entrada

Los ayudantes de entrada son una función de edición inteligente que ayuda a crear documentos XML válidos con rapidez. Mientras edita un documento, los ayudantes de entrada muestran opciones de edición estructural en base a la posición del cursor. Los ayudantes de entrada obtienen la información necesaria del esquema XML, del archivo DTD o de la hoja de estilos StyleVision Power Stylesheet subyacente. Por ejemplo, si está editando un documento XML, los elementos, atributos y entidades que se pueden insertar en la posición actual del cursor aparecerán en las ventanas de los ayudantes de entrada.

Tenga en cuenta los siguientes datos:

- Podrá encender y apagar la pantalla con los ayudantes de entrada con la siguiente opción del menú **Ventana | Ayudantes de entrada**.

© 2015 Altova GmbH

1.1.5 Ventana de resultados: Mensajes

La ventana Mensajes muestra información sobre las acciones realizadas en Authentic Desktop así como sobre errores y otros resultados. Por ejemplo, si se valida un documento XML y este resulta ser válido, aparece un mensaje de validación correcta en la ventana de mensajes. De lo contrario, aparece un mensaje que describe el error de validación. Recuerde que el mensaje de validación contiene hipervínculos (texto de vínculo negro) a los nodos y el contenido de los nodos del documento XML, así como hipervínculos (texto de vínculo azul) a la sección pertinente de la especificación que describe la regla en cuestión.

Validar carpetas y archivos en la ventana Proyecto

El comando **Validar** (del menú **XML**) se suele aplicar al documento activo. Sin embargo, también puede aplicar este comando a un archivo, una carpeta o un grupo de archivos del proyecto activo. Seleccione el archivo o la carpeta que desea validar en la ventana Proyecto y después haga clic en **XML | Validar el documento XML** o pulse la tecla **F8**. Los archivos del proyecto que no sean válidos se abren en la ventana principal y aparece un mensaje de error de tipo *El archivo no es válido*.

Nota: también puede realizar comprobaciones de formato XML (haciendo clic en el comando **Comprobar formato XML** o pulsando **F7**) en la ventana Proyecto.

1.1.6 Barra de menú, barras de herramientas y barra de estado

Barra de menú

La barra de menú (*ver ilustración*) contiene los menús de la aplicación. Tenga en cuenta estas normas generales:

- Si los comandos de un menú **no son pertinentes** en una vista o en una posición determinada del documento, entonces no están disponibles.
- Algunos comandos abren un submenú con una lista de opciones adicionales. Los comandos de menú que tienen submenús se identifican con una punta de flecha apuntando hacia la derecha, situada después del nombre del comando.
- Algunos comandos abren un cuadro de diálogo que solicita más información al usuario. Esta información es necesaria para llevar a cabo el comando seleccionado. Estos comandos se identifican con unos puntos suspensivos (...) situados después del nombre del comando.
- Para acceder a un comando de menú, haga clic en el nombre del menú y después en el comando. Si un comando incluye un submenú, el submenú se abre al pasar el puntero del mouse sobre el comando. A continuación haga clic en el comando correspondiente del submenú.
- Los menús se pueden abrir si se pulsa la combinación de teclas correspondiente. La combinación de teclas de cada menú es **Alt+*TECLA***, siendo ***TECLA*** la letra que aparece subrayada en el nombre del menú. Por ejemplo, para abrir el menú **Archivo** la combinación de teclas es **Alt+A**.
- Los comandos de menú (es decir, los comandos de cada menú) se pueden ejecutar si selecciona primero (i) el menú con su combinación de teclas (ver el punto anterior) y después (ii) la combinación de teclas del comando en sí (**Alt+*TECLA***, siendo ***TECLA*** la letra que aparece subrayada en el nombre del comando). Por ejemplo, para crear un archivo nuevo (**Archivo | Nuevo**), pulse **Alt+A** y después **Alt+N**.
- Algunos comandos de menú se pueden seleccionar directamente si presiona una tecla de **acceso rápido** o combinación de teclas (**Ctrl+*TECLA***). Puede identificar los comandos con accesos rápidos porque la tecla de acceso rápido o la combinación de teclas aparece después del nombre del comando. Por ejemplo, puede usar la combinación de teclas **Ctrl+N** para crear un archivo nuevo o la tecla de acceso rápido **F8** para validar un archivo XML. También puede crear sus propias teclas de acceso rápido en la pestaña Teclado del cuadro de diálogo "Personalizar" (**Herramientas | Personalizar**).

Barras de herramientas

Las barras de herramientas (*ver ilustración*) contienen iconos que funcionan como accesos rápidos para seleccionar comandos de menú. El nombre del comando aparece al pasar el puntero del mouse sobre el icono. Para ejecutar el comando, haga clic en el icono.

Los botones de las barras de herramientas se organizan en grupos. En el cuadro de diálogo **Herramientas | Personalizar | Barras de herramientas** puede indicar qué barras de herramientas desea activar. Estas opciones de configuración afectan a la vista actual. Para crear una configuración para otra vista, cambie a esa vista y vuelva a seleccionar las opciones correspondientes en el cuadro de diálogo **Herramientas | Personalizar | Barras de herramientas**. En la IGU también puede arrastrar barras de herramientas por sus controladores (o barras de título) y colocarlas en otra posición de la pantalla. Al hacer doble clic en el controlador la barra de herramientas se desacopla. Si hace doble clic otra vez en la barra de título, la barra de herramientas se acopla en la posición donde estuvo acoplada por última vez.

Barra de estado

© 2015 Altova GmbH

La barra de estado está situada en la parte inferior de la ventana de la aplicación (*ver ilustración*) y muestra (i) información de estado sobre la carga de archivos e (ii) información sobre los comandos de menú y los accesos rápidos a comandos de las barras de herramientas al pasar sobre estos el puntero del mouse.

1.2 El entorno de aplicación

En esta sección describimos varios aspectos de la aplicación que debe conocer antes de empezar a trabajar con ella. Esta sección le ayudará a familiarizarse con la aplicación para que pueda empezar a usarla con mayor confianza. Aquí encontrará información importante sobre opciones de configuración y personalización y sobre cómo cambiarlas.

Esta sección se divide en tres partes:

- *Configuración y personalización:* describe cómo definir importantes opciones de configuración y personalización.
- *Tutoriales, proyectos y ejemplos:* repasa la ubicación de los archivos incluidos con la aplicación que no pertenecen al programa.
- *Características y archivos de ayuda:* ofrece enlaces al sitio web de Altova, donde encontrará información sobre las funciones del producto, recursos de ayuda en varios formatos e información sobre otros productos de Altova.

© 2015 Altova GmbH

1.2.1 Configuración y personalización

Esta sección explica las opciones de configuración y personalización que se pueden elegir en Authentic Desktop.

Configuración

El cuadro de diálogo "Opciones" incluye varias pestañas donde se definen importantes opciones de configuración de Authentic Desktop . Repase las pestañas de este cuadro de diálogo y las diferentes opciones para familiarizarse con ellas.

Personalización

También puede personalizar varios aspectos de Authentic Desktop, incluido el aspecto de la IGU. Estas opciones de personalización están disponibles en el cuadro de diálogo "Personalizar" (comando de menú **Herramientas | Personalizar**).

Para una descripción más detallada de las opciones de personalización, consulte la sección **Referencia del usuario**.

1.2.2 Tutoriales, proyectos y ejemplos

El paquete de instalación de Authentic Desktop contiene tutoriales, proyectos y archivos de ejemplo.

Ubicación de los tutoriales, proyectos y archivos de ejemplo

Los tutoriales, proyectos y archivos de ejemplo de Authentic Desktop se instalan en esta carpeta:

© 2015 Altova GmbH

1.2.3 Características y archivos de ayuda

El sitio web de Altova www.altova.com contiene abundante información y gran cantidad de recursos relacionados con Authentic Desktop. A continuación mencionamos algunos ejemplos.

Lista de características de Authentic Desktop

El sitio web de Altova incluye una lista actualizada de las características de Authentic Desktop.

Ayuda de Authentic Desktop

La presente documentación es la ayuda creada por Altova para Authentic Desktop. Está disponible en el sistema integrado de ayuda de Authentic Desktop, al que puede tener acceso desde el menú **Ayuda** o pulsando la tecla **F1**. Además todos los manuales del usuario de los productos de Altova están disponibles en estos formatos:

- Manuales en línea en formato HTML, a los que puede tener acceso desde la página Soporte del sitio web de Altova
- Manuales en PDF para imprimir, que se pueden descargar desde el sitio web de Altova
- Copias impresas, que se pueden comprar en el sitio web de Altova

Opciones de soporte técnico

Si necesita más detalles sobre la información disponible en el *Manual del usuario* (la presente documentación) o tiene alguna pregunta sobre los productos de Altova, visite el Centro de soporte técnico del sitio web de Altova. Allí encontrará:

- Enlaces a nuestras páginas de preguntas frecuentes
- Foros de debate sobre los productos de Altova y temas relacionados con XML
- Formularios de soporte técnico para enviarnos solicitudes de soporte técnico (si cuenta con un paquete de soporte). Nuestro equipo de soporte técnico se encargará de responder a su solicitud.

Productos de Altova

El sitio web de Altova incluye una lista con todos los productos de Altova.

2 Tutorial de la vista Authentic

En la vista Authentic puede editar documentos XML en una interfaz gráfica WYSIWYG (*ver imagen siguiente*) como si se tratase de una aplicación de procesador de textos como Microsoft Word. De hecho, lo único que tiene que hacer es introducir los datos. No tiene que preocuparse por el formato del documento, puesto que este ya está definido en la hoja de estilos que controla la vista Authentic del documento XML. La hoja de estilos StyleVision Power Stylesheet (en adelante SPS) la crea un diseñador de hojas de estilos con Altova StyleVision.

Nanonull, Inc.
Location: US
Street: 119 Oakstreet, Suite 4876
City: Vereno
State & Zip: DC 29213
Phone: +1 (321) 555 5155 0
Fax: +1 (321) 555 5155 4
E-mail: office@nanonull.com
Vereno Office Summary: 4 departments, 15 employees.
The company was established in Vereno in 1995 as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed NanoSoft Development Suite in February 1999. Also in 1999, Nanonull increased its capital base with invesment from a consortium of private investment firms. The company has been expanding rapidly ever since.

Para editar un documento XML en la vista Authentic el usuario puede hacer dos cosas: (i) editar la estructura del documento (por ejemplo, agregar/eliminar párrafos y títulos) e (ii) introducir datos (el contenido de los párrafos y títulos, por ejemplo).

Este tutorial describe:

- Cómo se abre un documento XML en la vista Authentic. El principal requisito para editar documentos XML en la vista Authentic es que los documentos estén asociados a un archivo SPS.
- La interfaz de la vista Authentic y cómo funcionan los mecanismos de edición en general.
- Cómo se edita la estructura de un documento insertando y eliminando nodos.
- Cómo se introducen datos en un documento XML.
- Cómo se introducen valores de atributo desde el ayudantes de entrada Atributos y valores de entidades.
- Cómo se imprime el documento.

Recuerde que este tutorial es solo una introducción y, por tanto, es bastante simple y directo. Si desea más información consulte el material de referencia de la sección Interfaz de la vista

 © 2015 Altova GmbH

Authentic.

Material necesario para este tutorial
Todos los archivos necesarios para este tutorial se encuentran en la carpeta `...\Examples` de la carpeta de aplicación. Los archivos necesarios para este tutorial son:

- `NanonullOrg.xml` (el documento XML)
- `NanonullOrg.sps` (la hoja de estilos SPS a la que está vinculado el documento XML)
- `NanonullOrg.xsd` (el esquema XML en el que se basan y al que están vinculados el documento XML y la hoja de estilos SPS)
- `nanonull.gif` y `Altova_right_300.gif` (dos archivos de imagen usados en el tutorial)

Nota: en algunas secciones del tutorial le pediremos que mire el texto XML del documento XML (a diferencia de la vista Authentic del documento). Si la edición del producto de Altova que utiliza no incluye la vista Texto (si utiliza por ejemplo Authentic Desktop o Authentic Browser), entonces puede utilizar un **editor de texto** corriente como Wordpad o Notepad para ver el texto del documento XML.

Advertencia: le recomendamos que haga una copia del archivo `NanonullOrg.xml` y la utilice para este tutorial. Así podrá consultar siempre que quiera el documento original.

2.1 Abrir un documento en la vista Authentic

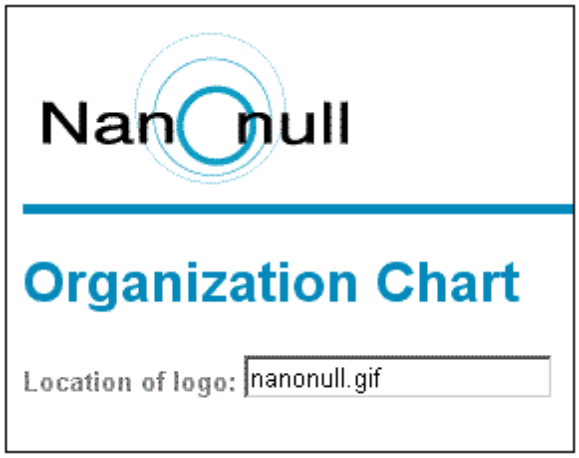
En la vista Authentic puede editar un documento XML ya existente o crear uno nuevo y editarlo. En esta sección del tutorial abrirá un documento XML en la vista Authentic y en las secciones siguientes aprenderá a editarlo. Esta sección también explica cómo se crea un documento XML nuevo para editarlo en la vista Authentic.

Abrir un documento XML ya existente

El archivo que abrirá es `NanonullOrg.xml`. Se encuentra en la carpeta `Examples` de su aplicación de Altova. Hay dos maneras de abrir el archivo:

- Haga clic en **Archivo | Abrir**, busque el archivo `NanonullOrg.xml` en el cuadro de diálogo que aparece y haga clic en **Abrir**.
- Utilice el explorador de Windows para localizar el archivo, haga clic con el botón secundario en el archivo y seleccione su producto de Altova en el submenú **Abrir con**.

El archivo `NanonullOrg.xml` se abre directamente en la vista Authentic (*imagen siguiente).*



Nota: el archivo que define y controla cómo se presenta el documento XML en la vista Authentic es el archivo SPS. Si no existe un SPS, el documento no se puede mostrar en la vista Authentic.

Crear un documento XML basado en una SPS

También puede crear un documento XML nuevo basado en una SPS. Hay dos maneras de hacerlo: con el comando de menú **Archivo | Nuevo** o con el comando de menú **Authentic | Documento nuevo**. En ambos casos es necesario seleccionar un archivo SPS.

Opción 1: Archivo | Nuevo

1. Seleccione **Archivo | Nuevo** y en el cuadro de diálogo "Crear documento nuevo" seleccione el tipo de archivo XML.
2. Haga clic en el botón **Seleccione una hoja de estilos de StyleVision** y busque el archivo SPS.

Opción 2: Authentic | Documento nuevo

1. Seleccione **Authentic | Documento nuevo**.
2. En el diálogo "Crear documento nuevo" busque el archivo SPS.

© 2015 Altova GmbH

Si se asignó un archivo de plantilla XML al SPS, entonces los datos de la plantilla XML se usarán como datos de partida del documento XML que se crea en la vista Authentic.

2.2 La interfaz de la vista Authentic

La interfaz de edición de la vista Authentic está compuesta por una ventana principal, en la que se introducen y editan los datos del documento, y tres ayudantes de entrada. Editar documentos en la vista Authentic es fácil. Si desea ver el marcado del documento, active las etiquetas de marcado. Después puede empezar a escribir en el contenido del documento. Para cambiar la estructura del documento puede usar el menú contextual o el ayudante de entrada Elementos.

Mostrar las etiquetas de los nodos XML (marcado del documento)

Un documento XML es básicamente una jerarquía de nodos. Por ejemplo:

```
<DocumentRoot>
   <Person id="ABC001">
       <Name>Alpha Beta</Name>
       <Address>Some Address</Address>
       <Tel>1234567</Tel>
   </Person>
</DocumentRoot>
```

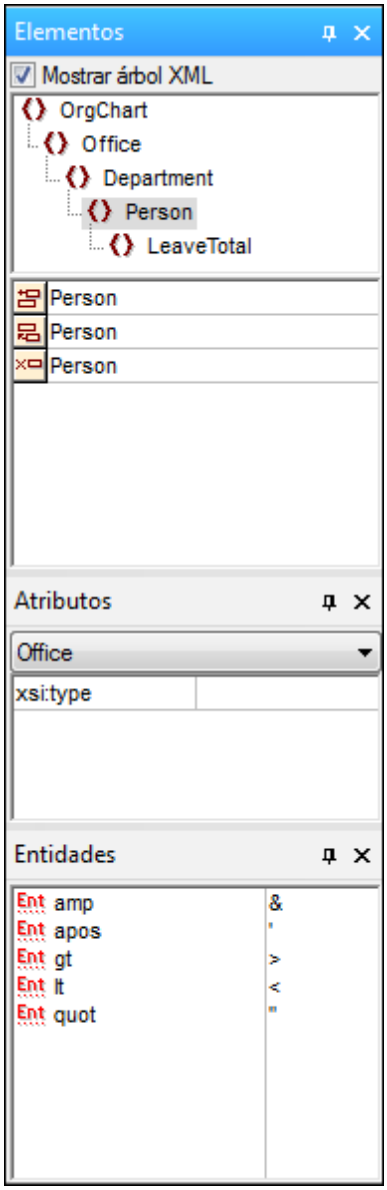
La opción predeterminada de la vista Authentic es no mostrar las etiquetas de los nodos. Para activar las etiquetas de los nodos, seleccione el comando de menú **Authentic | Mostrar marcado en tamaño grande** (o haga clic en el icono de la barra de herramientas). Las etiquetas en tamaño grande muestran el nombre del nodo. También puede ver el marcado en tamaño pequeño (las etiquetas no incluyen el nombre del nodo) o como marcado mixto (definido por el diseñador de la hoja de estilos. En el marcado mixto predeterminado todas las etiquetas están desactivadas).

Puede ver el texto del documento XML en la vista Texto de su producto de Altova o en un editor de texto.

Ayudantes de entrada

Hay tres ayudantes de entrada en la interfaz de la vista Authentic (*imagen siguiente*), situados a la derecha de la ventana de la aplicación. Son los ayudantes de entrada Elementos, Atributos y Entidades.

© 2015 Altova GmbH



Ayudante de entrada Elementos: este ayudante de entrada muestra los elementos que se pueden insertar y quitar dependiendo de la posición del cursor o del texto que está seleccionado en la ventana principal. Tenga en cuenta que el ayudante de entrada es contextual y su contenido cambia dependiendo de la posición del cursor o del texto que está seleccionado. Además, cuando se selecciona un nodo del árbol XML del ayudante de entrada, los elementos relacionados con ese nodo aparecen en el ayudante de entrada. Para ver el árbol XML en el ayudante de entrada Elementos, marque la casilla "Mostrar árbol XML" situada en la parte superior del ayudante (*ver imagen anterior*). El árbol XML muestra la jerarquía de nodos a partir del elemento de nivel superior y hasta llegar al nodo que está seleccionado en la ventana principal.

Ayudante de entrada Atributos: este ayudante de entrada muestra los atributos del elemento que está seleccionado en la ventana principal. Los valores de los atributos se pueden introducir y editar en este ayudante. El ayudante también incluye un cuadro combinado donde puede seleccionar todos los elementos de la jerarquía, desde el elemento de nivel superior hasta el

elemento seleccionado. Al seleccionar un elemento en el cuadro combinado aparecen los atributos de ese elemento en el ayudante de entrada, donde pueden editarse.

Ayudante de entrada Entidades: este ayudante de entrada no es contextual y muestra todas las entidades que se declararon para el documento. Al hacer doble clic en una entidad, esta se inserta en la posición del cursor. Para más información sobre cómo añadir entidades en un documento, consulte la sección Interfaz de la vista Authentic.

Menú contextual

Al hacer clic con el botón secundario en un documento abierto en la vista Authentic aparece un menú contextual con opciones relacionadas con la posición (el nodo) del documento donde se hizo clic. El menú contextual incluye comandos para:

- Insertar nodos en la posición del nodo, antes o después. Los submenús incluyen todos los nodos permitidos en la posición donde se hizo clic.
- Quitar el nodo seleccionado (si el esquema lo permite) o los elementos antecesores que se puedan quitar. Los nodos que se pueden quitar (de acuerdo con el esquema) aparecen en un submenú.
- Insertar entidades y secciones CDATA. Las entidades declaradas para el documento aparecen en un submenú. Las secciones CDATA sólo se pueden insertar dentro de texto.
- Cortar, copiar, pegar (como texto o como XML) y eliminar contenido del documento.

Nota: para más información sobre la interfaz, consulte la sección Interfaz de la vista Authentic.

© 2015 Altova GmbH

2.3 Operaciones con nodos

En la vista Authentic de un documento XML existen dos tipos principales de nodos: **nodos de elemento** y **nodos de atributo**. Estos nodos están marcados con etiquetas, que puede activar si lo desea. El documento también puede incluir otros nodos, como nodos de texto (que no están marcados) y nodos de sección CDATA (que están marcados para separarlos del texto que los rodea).

Las operaciones de nodo descritas en este apartado se refieren únicamente a los nodos de elemento y de atributo. Recomendamos que tenga activado el marcado en tamaño grande mientras prueba las operaciones descritas en este apartado.

Nota: Es importante recordar que antes y después del elemento seleccionado sólo se pueden insertar **elementos del mismo nivel o de nivel superior**. Por ejemplo, los **elementos del mismo nivel** de un elemento párrafo serían otros elementos párrafos, pero también listas, una tabla, una imagen, etc. Los elementos del mismo nivel pueden aparecer antes o después de un elemento. Los elementos de nivel superior son los elementos **antecesores** y los elementos del mismo nivel de los antecesores. Por ejemplo, los elementos antecesores de un elemento párrafo serían un elemento sección, capítulo, artículo, etc. Un párrafo de un archivo XML válido ya tendría elementos antecesores. Por tanto, al añadir un elemento de nivel superior en la vista Authentic, el elemento nuevo que se añade se crea al mismo nivel que el antecesor correspondiente. Por ejemplo, si se inserta una sección después de un párrafo, se crea como elemento del mismo nivel que la sección en la que se encuentra dicho párrafo.

Realizar operaciones con nodos

Para realizar operaciones con nodos seleccione un comando del menú contextual o haga clic en la opción correspondiente del ayudante de entrada Elementos. En algunos casos se pueden añadir elementos o atributos haciendo clic en el hipervínculo Agregar nodo de la vista Authentic del documento. Hay casos especiales en los que los elementos se definen como párrafos o elementos de lista. Si coloca el cursor dentro de dichos elementos y pulsa la tecla Entrar, se crea un elemento del mismo nivel de ese tipo. Este apartado también describe cómo se crean y eliminan nodos con los mecanismos Aplicar elemento, Quitar nodo y Borrar elemento.

Insertar elementos

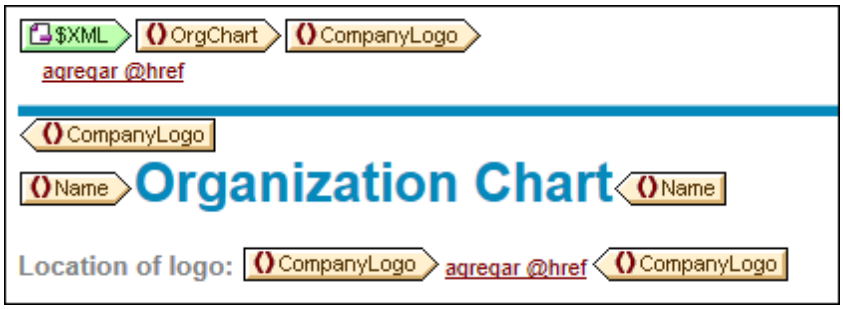
Puede insertar elementos en estas posiciones:

- Dentro del nodo de elemento, en la posición donde está situado el cursor. Los elementos que se pueden insertar en la posición seleccionada aparecen en un submenú del comando **Insertar** del menú contextual. En el ayudante de entrada Elementos los elementos que se pueden insertar en la posición seleccionada llevan el icono . En el documento `NanonullOrg.xml` coloque el cursor dentro del elemento `para` y cree los elementos `bold` e `italic` usando tanto el menú contextual como el ayudante de entrada Elementos.
- Antes o después del elemento seleccionado o de sus antecesores, si el esquema lo permite. Seleccione el elemento que desea insertar en los submenús del menú contextual. En el ayudante de entrada Elementos los elementos que se pueden insertar antes o después del elemento seleccionado llevan los iconos y respectivamente. Recuerde que desde el ayudante de entrada Elementos no se pueden insertar elementos antes ni después del elemento antecesor, sino antes o después del elemento seleccionado. Pruebe este comando primero dentro del elemento `para` y después dentro

de la tabla que incluye los empleados de la compañía.

Hipervínculo Agregar nodo

Si un elemento o atributo está incluido en el diseño pero no está presente en el documento XML, entonces aparece el hipervínculo Agregar nodo en la posición del documento donde se especifica dicho nodo. Para ver este hipervínculo, vaya a la línea que tiene el texto *Location of logo*, seleccione el nodo `@href` del elemento `CompanyLogo` y elimínelo (pulsando la tecla **Suprimir**). Paso seguido aparece el hipervínculo `agregar @href` dentro del elemento `CompanyLogo` que acaba de cambiar (*ver imagen siguiente*). Al hacer clic en el hipervínculo se añade el nodo `@href` al documento XML. El cuadro de texto situado dentro de las etiquetas de `@href` aparece porque el diseño indica que el nodo `@href` se debe añadir de esta manera. Es decir, todavía tiene que introducir el valor (o el contenido) del nodo `@href`. Escriba `nanonull.gif` en el cuadro de texto.



Si el modelo de contenido de un elemento es ambiguo, por ejemplo, si especifica que una secuencia de elementos secundarios puede aparecer en cualquier orden, entonces aparece el hipervínculo `agregar...`. (sin el nombre del nodo). Al hacer clic en este hipervínculo aparece una lista de elementos que se pueden insertar.

Nota: el hipervínculo `agregar Nodo` aparece directamente en la plantilla del documento. No hay una opción equivalente ni en el menú contextual ni en el ayudante de entrada Elementos.

Crear elementos nuevos con la tecla Entrar

Hay casos en los que el diseñador de la hoja de estilos dio formato de párrafo o elemento de lista a un nodo. Si sitúa el cursor dentro de un nodo de este tipo y pulsa la tecla **Entrar**, entonces se inserta otro nodo del mismo tipo después del nodo seleccionado. Pruebe este mecanismo en el documento `NanonullOrg.xml`. Para ello sitúe el cursor al final de un nodo `para` (es decir, justo antes de su etiqueta final) y pulse la tecla **Entrar**.

Aplicar elementos

En los elementos de contenido mixto (es decir los que incluyen tanto texto como elementos secundarios) puede seleccionar parte del texto y aplicarle un elemento secundario. Como resultado el texto pasa a ser el contenido del elemento que se aplicó. Para aplicar elementos, seleccione **Aplicar** en el menú contextual y después seleccione uno de los elementos disponibles. (Si no se puede aplicar ningún elemento al texto seleccionado, entonces el comando **Aplicar** no aparece en el menú contextual.) En el ayudante de entrada Elementos los elementos que se pueden aplicar al texto seleccionado llevan el icono . En el documento `NanonullOrg.xml` seleccione parte del texto del contenido del elemento `para` y pruebe esta operación, aplicando los elementos `bold` e `italic`.

© 2015 Altova GmbH

El diseñador de la hoja de estilos también puede crear un icono en la barra de herramientas para aplicar un elemento. Es el caso del documento `NanonullOrg.xml`, cuyos elementos `bold` e `italic` se pueden aplicar con solo hacer clic en los iconos **B** e ***I*** de la barra de herramientas de Authentic.

Quitar nodos

Puede quitar un nodo siempre y cuando al quitarlo el documento siga siendo válido. Cuando se quita un nodo se elimina también su contenido. Para quitar un nodo seleccione el comando **Quitar** del menú contextual. Cuando el comando **Quitar** está resaltado, aparece un submenú que contiene todos los nodos que se pueden quitar, empezando por el nodo seleccionado hasta llegar al nodo de nivel superior del documento. Para seleccionar el nodo que desea quitar, coloque el cursor dentro del nodo o resalte el nodo (o parte del nodo). En el ayudante de entrada Elementos los nodos que se pueden quitar llevan el icono . También puede hacer clic en un nodo y pulsar la tecla **Suprimir**. Pruebe esta operación en el documento `NanonullOrg.xml` y quite varios nodos usando los diferentes mecanismos descritos en este párrafo. Para deshacer los cambios pulse **Ctrl+Z**.

Borrar elementos

Puede borrar los elementos secundarios de elementos de contenido mixto (elementos que tienen texto y elementos secundarios). Para borrar el elemento entero, seleccione el nodo o coloque el cursor dentro del nodo. Para borrar un fragmento de texto, resalte el fragmento de texto. Cuando termine de seleccionar el nodo o fragmento de texto que desea borrar, seleccione la opción **Borrar** del menú contextual y después seleccione el elemento que desea borrar. En el ayudante de entrada Elementos los elementos que se pueden borrar en la posición seleccionada llevan el icono (si la selección se hizo colocando el cursor en un punto de inserción) o el icono (si seleccionó un intervalo). Pruebe este tipo de operación en el documento `NanonullOrg.xml`. Por ejemplo, intente borrar los elementos secundarios `bold` e `italic` del nodo `para` (que tiene contenido mixto).

Tablas y estructura de las tablas

En la vista Authentic hay dos tipos de tablas:

- *Tablas SPS (estáticas y dinámicas).* La estructura general de una tabla SPS la define el diseñador de la hoja de estilos. Dentro de la estructura general de la tabla, los únicos cambios que se pueden realizar son cambios de contenido. Por ejemplo, puede añadir filas nuevas en una tabla SPS dinámica.
- *Tablas XML.* Puede decidir si presenta el contenido de un nodo en forma de tabla. Si el diseñador de la hoja de estilos habilitó la posibilidad de crear este nodo en forma de tabla, entonces puede definir su estructura y editar su contenido. Las tablas XML se explican en detalle en el apartado Tablas en la vista Authentic.

2.4 Introducir datos en la vista Authentic

En la vista Authentic los datos se introducen en el documento XML desde la ventana principal. Los valores de los atributos también se pueden introducir en el ayudante de entrada Atributos. Los datos se pueden introducir de dos maneras: (i) escribiendo el texto o (ii) mediante un dispositivo de entrada de datos, que se asigna a una entrada de texto predefinida.

Agregar contenido de texto
En la ventana principal de la vista Authentic puede introducir el contenido de los elementos y el valor de los atributos directamente en forma de texto. Para insertar contenido, sitúe el cursor en la posición deseada y teclee el texto. También puede copiar texto y pegarlo en el documento. También puede usar las teclas **Mayús** y **Suprimir**. Por ejemplo, puede resaltar el texto que desea editar y teclear el texto nuevo con la tecla **Mayús** activada.

Por ejemplo, para cambiar el nombre de la compañía, en el campo `Name` de `Office`, sitúe el cursor después de *Nanonull* y teclee *USA* para cambiar el nombre de la compañía a *Nanonull USA, Inc*.

Si el texto se puede editar, entonces podrá situar el cursor en el texto y resaltarlo. De lo contrario no podrá resaltar el texto con el cursor. Por ejemplo, intente cambiar el **nombre de los campos** (no su valor) *Street*, *City* o *State/Zip* del bloque `Address`. Verá que no puede colocar el cursor en este texto porque no es contenido XML, sino que viene dado por la hoja de estilos StyleVision Power Stylesheet.

Insertar caracteres especiales y entidades

En la vista Authentic los caracteres especiales y las entidades se gestionan de manera especial:

- *Caracteres especiales que se usan para marcado XML (&, ', >, <, ").* Estos caracteres son entidades integradas y se puede introducir en el documento con solo hacer doble clic en ellos en el ayudante de entrada Entidades. Si estos caracteres aparecen con frecuencia (en fragmentos de código, por ejemplo) puede incluirlos en secciones CDATA. Para introducir una sección CDATA, haga clic con el botón secundario en la posición elegida y seleccione **Insertar sección CDATA** en el menú contextual. El procesador XML ignora todos los caracteres de marcado que están dentro de secciones CDATA. Si desea introducir un carácter especial dentro de una sección CDATA, introduzca el carácter, no su referencia de entidad.
- *Caracteres especiales que no se pueden introducir con el teclado.* Para introducir estos caracteres cópielos y péguelos en la posición deseada.
- Puede definir como entidades las *cadenas de texto que se usan frecuentemente*. Las entidades aparecen en el ayudante de entrada Entidades. Para insertar una entidad sitúe el cursor en la posición elegida y haga doble clic en la entidad en el ayudante de entrada. Esta función es muy útil porque el valor de la cadena de texto se guarda en una sola ubicación y si necesita cambiar el valor de la cadena, basta con cambiar la definición de entidad.

© 2015 Altova GmbH

Nota: en la vista Authentic es fácil pasar por alto los elementos vacíos si el marcado XML está oculto. Por eso recomendamos que active el marcado en tamaño grande o pequeño.

Pruebe a insertar caracteres especiales y entidades tal y como acabamos de explicar.

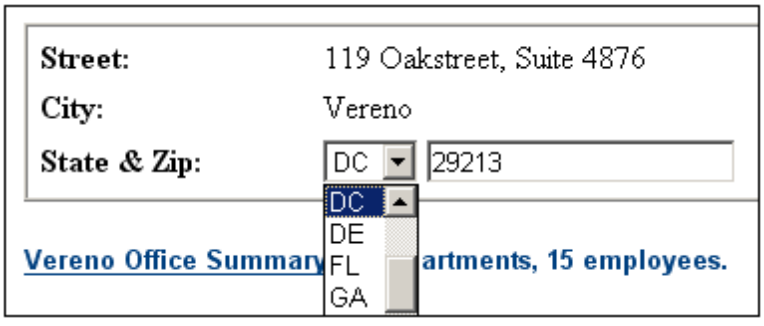
Agregar contenido mediante dispositivos de entrada de datos

Hasta ahora aprendió a introducir datos tecleando el texto directamente en el documento. En la vista Authentic también puede introducir **contenido en los elementos** (o valores de atributo) mediante dispositivos de entrada de datos.

A continuación mostramos una lista de los dispositivos de entrada de datos de la vista Authentic, con una breve explicación de cada uno de ellos.

Dispositivo	Datos en el archivo XML
Campo de entrada (cuadro de texto)	Texto introducido por el usuario
Campo de entrada multilínea	Texto introducido por el usuario
Cuadro combinado	La opción elegida por el usuario tiene asignado un valor
Casilla de verificación	La opción elegida por el usuario tiene asignado un valor
Botón de opción	La opción elegida por el usuario tiene asignado un valor
Botón	La opción elegida por el usuario tiene asignado un valor

Por ejemplo, en la tabla estática que contiene los campos de dirección (*imagen siguiente*), hay dos dispositivos de entrada de datos: un campo de entrada para el campo *Zip* y un cuadro combinado para el campo *State*. Los valores que introduzca en el campo de entrada se introducen en el elemento correspondiente del documento XML. En el cuadro combinado cada opción tiene asignado ya un valor y en el documento XML se introduce el valor de la opción seleccionada por el usuario.



El texto XML de la vista Authentic de la imagen anterior sería:

```
<Address>
```

```
      <ipo:street>119 Oakstreet, Suite 4876</ipo:street>
      <ipo:city>Vereno</ipo:city>
      <ipo:state>DC</ipo:state>
      <ipo:zip>29213</ipo:zip>
    </Address>
```

Observe que la opción `DC` del cuadro combinado tiene asignado el valor `DC`. Por el contrario, el valor del campo `Zip` lo introduce el usuario libremente.

© 2015 Altova GmbH

2.5 Introducir valores de atributo

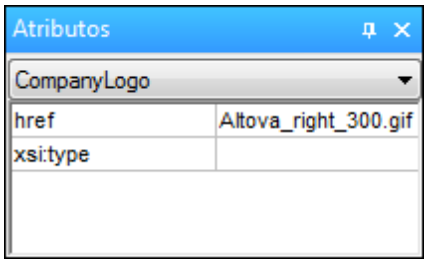
Un atributo es una propiedad de un elemento y un elemento puede tener varios atributos. Los atributos tienen valores. A veces puede ser necesario introducir datos XML como valor del atributo. En la vista Authentic hay dos maneras de introducir valores de atributo:

- en la ventana principal, si así lo decidió el diseñador de la hoja de estilos y
- en el ayudante de entrada Atributos.

Valores de atributo en la ventana principal
Para introducir el valor de un atributo puede escribir texto, escribir texto en un campo de entrada o seleccionar una opción de entrada que tiene asignado un valor XML. Para más información consulte el apartado Introducir datos en la vista Authentic.

Valores de atributo en el ayudante de entrada Atributos
También puede introducir un valor para un atributo o cambiar el valor de un atributo desde el ayudante de entrada Atributos. Primero seleccione el nodo de atributo en la vista Authentic. Después introduzca o edite el valor del atributo en el ayudante de entrada Atributos. En el documento `NanonullOrg.xml` la ubicación del logotipo viene dada por el valor del atributo `href` del elemento `CompanyLogo`. Para cambiar el logotipo de la compañía utilizado en el documento siga estos pasos:

1. Haga clic en la etiqueta `CompanyLogo` para seleccionar el elemento `CompanyLogo`. Los atributos del elemento `CompanyLogo` aparecen en el ayudante de entrada Atributos.
2. En el ayudante de entrada Atributos cambie el valor `nanonull.gif` del atributo `href` por `Altova_right_300.gif` (una imagen de la carpeta `Examples`).

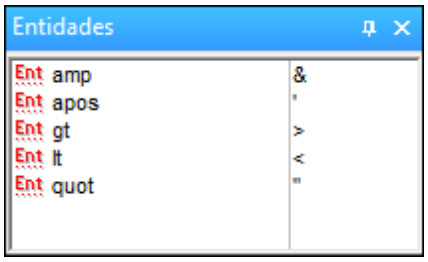


Como resultado el logotipo Nanonull desaparece y lo sustituye el logotipo de Altova.

Nota: en el ayudante de entrada Atributos no se pueden introducir entidades.

2.6 Añadir entidades

En la vista Authentic las entidades suelen ser datos XML (aunque no siempre), como un solo carácter, una cadena de texto o incluso un fragmento de un documento XML. Las entidades también pueden ser archivos binarios, como archivos de imagen por ejemplo. Todas las entidades que se pueden usar en el documento aparecen en el ayudante de entrada Entidades (*imagen siguiente*). Para insertar una entidad, sitúe el cursor en la posición deseada y haga doble clic en la entidad en el ayudante de entrada Entidades. Recuerde que en el ayudante de entrada Atributos no se pueden insertar entidades.



El carácter et (&) es de especial importancia en XML (al igual que los caracteres ', >, < y "). Para insertar estos caracteres, se utilizan entidades y así no se confunden con caracteres XML. Estos caracteres son entidades predeterminadas en la vista Authentic.

En el documento `NanonullOrg.xml` cambie el puesto de (*Title*) de Joe Martin (del departamento de Marketing) por *Marketing Manager Europe & Asia*. Para ello siga estos pasos:

1. Sitúe el cursor en la posición donde debe insertarse el carácter et (&).
2. En el ayudante de entrada Entidades, haga doble clic en la entidad `amp`. Como resultado se inserta el carácter et (&).

Nota: el ayudante de entrada Entidades no es contextual y muestra todas las entidades disponibles independientemente de la posición del cursor. Esto no significa que las entidades se puedan introducir en cualquier parte del documento. Por ello, después de insertar una entidad, es muy importante validar el documento: **XML | Validar** (**F8**).

Definir entidades propias

Si lo desea también puede editar sus propias entidades. Para aprender a hacerlo consulte el apartado Definir entidades en vista Authentic.

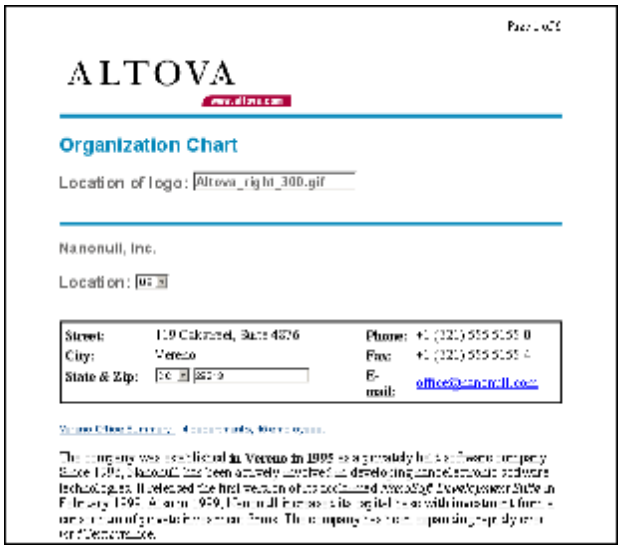
© 2015 Altova GmbH

2.7 Imprimir el documento

Si lo desea puede imprimir la vista Authentic de su documento.

Para imprimir la vista Authentic de `NanonullOrg.xml` siga estos pasos:

1. Cambie al modo **Ocultar marcado** (es necesario desactivar el marcado si no quiere que se imprima).
2. Seleccione **Archivo | Vista previa de impresión** para obtener una vista previa de todas las páginas. A continuación aparece parte de la vista previa, reducida un 50%.



   Observe que el formato de la página es idéntico al que tiene en la vista Authentic.
3. Para imprimir el archivo, haga clic en **Archivo | Imprimir**.

Recuerde que también puede imprimir el documento con el marcado XML. Para ello, cambie la vista Authentic al modo **Mostrar marcado en tamaño grande** o **Mostrar marcado en tamaño pequeño** y después imprima el documento.

3 Interfaz de la vista Authentic

La vista Authentic se habilita haciendo clic en la pestaña Authentic del documento activo. Si no se asignó un archivo SPS al documento XML, se le pedirá que asigne uno. Para asignar un archivo SPS utilice el comando **Authentic | Asignar hoja de estilos de StyleVision**.

Esta sección incluye:

- Una presentación de la interfaz gráfica.
- Una descripción de los iconos de la barra de herramientas propios de la vista Authentic.
- Una descripción de los modos de visualización disponibles en la ventana principal de la vista Authentic.
- Una descripción de los ayudante de entrada e información sobre cómo usarlos.
- Una descripción de los menús contextuales disponibles en varios puntos de la vista Authentic del documento XML

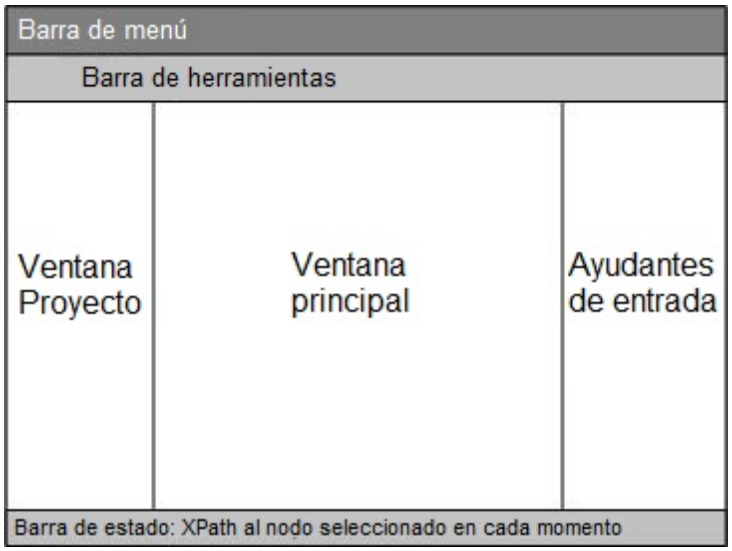
Para más información sobre la vista Authentic consulte estos recursos:

- El tutorial de la vista Authentic, que explica cómo usar la interfaz de la vista Authentic. Este tutorial está disponible en la documentación de Altova XMLSpy.
- La *Referencia del usuario* de su producto, que incluye una descripción detallada de los comandos de menú de la vista Authentic.

© 2015 Altova GmbH

3.1 La interfaz gráfica del usuario (IGU)

La vista Authentic tiene una barra de menú y una barra de herramientas situadas en la parte superior de la ventana y tres secciones: la ventana Proyecto, la ventana principal y la ventana de los ayudantes de entrada. La imagen que aparece a continuación muestra todos estos componentes.



Barra de menú
Los menús de la barra de menú se describen en detalle en la *Referencia del usuario* del producto.

Barra de herramientas
Los símbolos e iconos de la barra de herramientas se describen en la sección iconos de la barra de herramientas de la vista Authentic.

Ventana Proyecto
Puede agrupar en un proyecto todos los archivos XML, XSL, XSD y de entidad. Para crear o modificar la lista de archivos del proyecto, use los comandos del menú Proyecto (descritos en la *Referencia del usuario* de su producto). La lista de archivos del proyecto aparece en la ventana Proyecto. Para tener acceso a los archivos de la ventana Proyecto, haga doble clic en el archivo que le interese.

Ventana principal
Esta es la ventana donde se visualiza y edita el documento XML . Para más información, consulte la sección ventana principal de la vista Authentic.

Ayudantes de entrada
La vista incluye tres ayudantes de entrada: Elementos, Atributos y Entidades. Las opciones de entrada de los ayudantes Elementos y Atributos son contextuales, es decir, dependen de la posición del cursor en el documento. Para introducir un elemento o entidad en el documento, haga doble clic en ellos en su ayudante de entrada. El valor de un atributo se introduce en el campo de valor del atributo en el ayudante de entrada Atributos. Para más información consulte la sección ayudantes de entrada de la vista Authentic.

Barra de estado

La barra de estado muestra la XPath al nodo que esté seleccionado en cada momento.

Menús contextuales
Los menús contextuales son los menús que aparecen al hacer clic con el botón secundario en la ventana principal. Los comandos son comandos de edición contextuales, es decir, permiten manipular la estructura y el contenido del nodo seleccionado. Por ejemplo, puede insertar, anexar o eliminar un nodo, añadir entidades o cortar y pegar contenido.

© 2015 Altova GmbH

3.2 Iconos de la barra de herramientas de la vista Authentic

Los iconos de la barra de herramientas de la vista Authentic son básicamente accesos rápidos a los comandos. Algunos iconos le sonarán de otras aplicaciones de Windows o de otros productos de Altova, mientras que otros le serán totalmente nuevos. Esta sección describe los iconos propios y exclusivos de la vista Authentic.

Mostrar u ocultar marcado XML
En la vista Authentic puede elegir si se presentan las etiquetas de todos los elementos o atributos XML, de algunos o de ninguno, ya sea con el nombre de la etiqueta (marcado de tamaño grande) o sin el nombre de la etiqueta (marcado de tamaño pequeño). Los cuatro iconos de marcado de la barra de herramientas se corresponden con cuatro comandos del menú **Authentic**.

Ocultar marcado. Se ocultan todas las etiquetas XML excepto las que se contrajeran. Si pulsa el icono *Ocultar marcado* y después hace doble clic en una etiqueta contraída (la manera normal de expandirla), entonces se mostrará el contenido del nodo pero las etiquetas se ocultarán.

Mostrar marcado en tamaño pequeño. Las etiquetas de los atributos o elementos XML se muestran sin el nombre de etiqueta.

Mostrar marcado en tamaño grande. Las etiquetas de los atributos o elementos XML se muestran con el nombre de etiqueta.

Mostrar marcado mixto. En la StyleVision Power Stylesheet se puede especificar cómo se debe mostrar cada elemento o atributo XML (con marcado grande o pequeño) o si no se debe mostrar. Esto recibe el nombre de marcado mixto porque algunos elementos se mostrarán con marcado y otros no. Por tanto, en el modo de marcado mixto el usuario de la vista Authentic ve un tipo de marcado personalizado. No obstante, es necesario decir que quien configura estas opciones es la persona que diseña la StyleVision Power Stylesheet. El usuario de la vista Authentic no puede configurar estas opciones.

Editar estructuras de tablas dinámicas
Las filas de una **tabla dinámica SPS** son básicamente repeticiones de una estructura de datos. Cada fila representa una repetición de un único elemento. Cada fila, por tanto, tiene la misma subestructura XML que la fila siguiente.

Los comandos de edición de la tabla dinámica permiten manipular las filas de una tabla dinámica SPS. Es decir, puede cambiar el número de veces que aparecen los elementos y el orden en el que aparecen. Sin embargo, no puede editar las columnas de una tabla dinámica SPS porque eso supondría cambiar la subestructura de las diferentes repeticiones del elemento.

Los iconos de edición de tablas dinámicas están en la barra de herramientas y se corresponden con los comandos del menú **Authentic**.

Anexar una fila a la tabla

Insertar una fila en la tabla

Duplicar la fila actual (es decir, el contenido de la celda se duplica)

Subir la fila actual

Bajar la fila actual

Eliminar la fila actual

Nota: estos comandos corresponden sólo a las **tablas dinámicas SPS**. No se deben usar dentro de tablas SPS estáticas. Los distintos tipos de tablas utilizados en la vista Authentic se describen en la sección Usar tablas en la vista Authentic de la presente documentación.

Crear y editar tablas XML

Si desea presentar sus datos en forma de tabla, puede insertar sus propias tablas. Estas tablas se insertan como tablas XML. Puede modificar la estructura de la tabla XML y dar formato a la tabla. Los iconos para crear y editar tablas XML están disponibles en la barra de herramientas (*ver imagen siguiente*). Estos iconos se describen en detalle en la sección Iconos de edición de tablas XML.

Los comandos de estos iconos **no están disponibles como opciones de menú**. Recuerde también que para poder usar tablas XML, esta función debe estar habilitada y configurada correctamente en la StyleVision Power Stylesheet.

Para más información sobre los tipos de tablas utilizados en la vista Authentic y sobre cómo crear y editar tablas XML, consulte la sección Usar tablas en la vista Authentic.

Iconos de formato de texto

Para dar formato al texto en la vista Authentic es necesario aplicar el elemento o atributo XML que contiene el formato deseado. Si el formato deseado ya se definió, el diseñador de la StyleVision Power Stylesheet puede incluir iconos para dicho formato en la barra de herramientas de la vista Authentic. Para dar formato al texto, seleccione el texto y pulse el icono correspondiente.

Iconos de navegación por filas de base de datos

Estos iconos significan, de izquierda a derecha: *Ir al primer registro de la BD*,

Ir al registro anterior, Abrir el cuadro de diálogo *Ir al registro número*, *Ir al registro siguiente* e *Ir al último registro*.

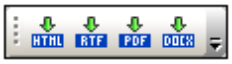
Este icono abre el cuadro de diálogo *Editar la consulta de base de datos*, donde puede escribir una consulta. La vista Authentic mostrará los registros de la consulta.

Edición de bases de datos XML

El comando **Seleccionar y editar una fila nueva con datos XML** permite seleccionar una fila nueva de la tabla pertinente en una base de datos XML, como IBM DB2. Esta fila aparece en la vista Authentic, donde se puede editar. Después puede guardar la fila otra vez en la base de datos.

Botones de la barra de herramientas para formularios XML portátiles (PXF)

La vista Authentic de Authentic Desktop incluye estos botones en la barra de herramientas para archivos PXF:



Al hacer clic en cada uno de estos botones, se genera un archivo de salida en formato HTML, RTF, PDF y DocX respectivamente.

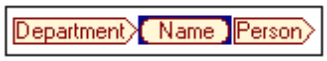
Estos botones se habilitan cuando se abre un archivo PXF en la vista Authentic y de acuerdo con la configuración del archivo PXF. Por ejemplo, si el archivo PXF se configuró para contener una hoja de estilos XSLT para formatos de salida HTML y RTF, entonces se habilitarán los botones HTML y RTF, pero los botones PDF y DocX estarán deshabilitados.

3.3 Ventana principal de la vista Authentic

La vista Authentic ofrece cuatro modos de visualización: marcado grande, marcado pequeño, marcado mixto y marcado oculto. Estos modos permiten visualizar la información de marcado del documento de diferentes maneras. Para cambiar de un modo a otro, utilice los comandos del menú Authentic o los iconos de la barra de herramientas (ver el apartado anterior Iconos de la barra de herramientas de la vista Authentic).

Marcado de tamaño grande
Este modo muestra las etiquetas iniciales y finales de los elementos y atributos con los nombres del elemento o atributo:

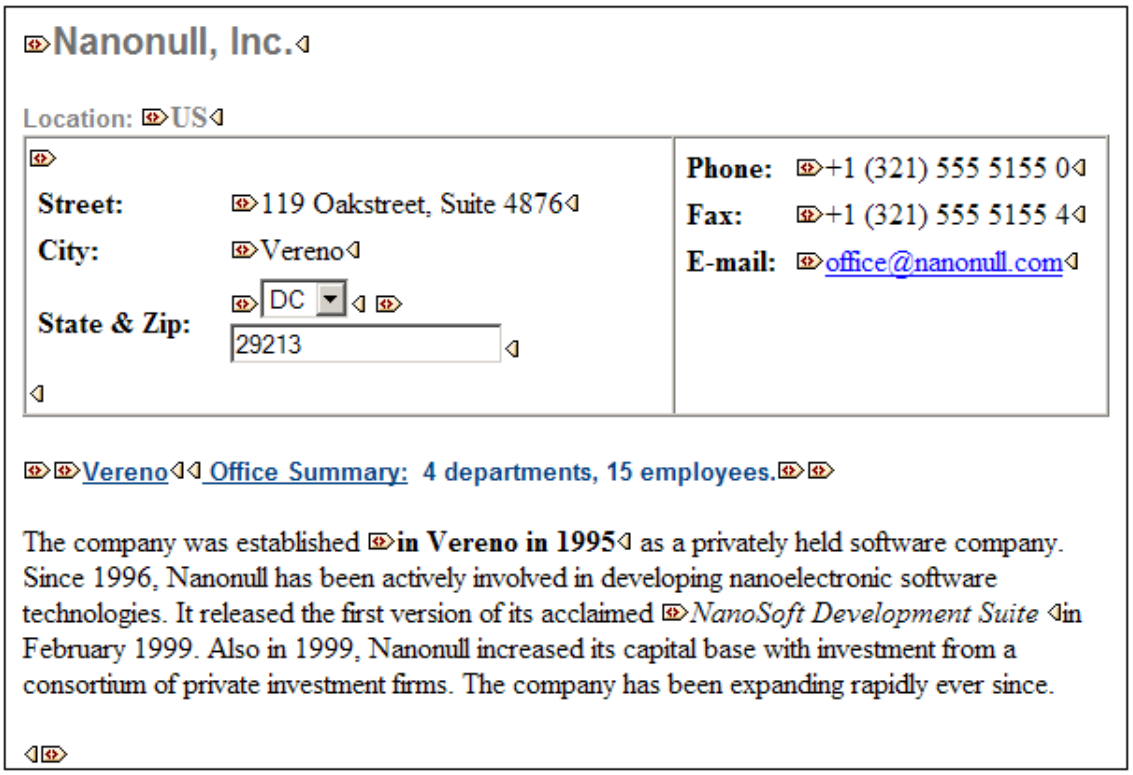
El elemento `Name` de la imagen anterior está **expandido**, es decir, se muestra tanto las etiquetas iniciales y finales del elemento como su contenido. Los elementos y atributos se pueden **contraer**. Para ello haga doble clic en la etiqueta inicial o en la final. Para volver a expandir el elemento o atributo contraído, haga doble clic en la etiqueta contraída.



En el modo de marcado grande, los atributos se identifican porque las etiquetas iniciales y finales llevan el signo =:

Marcado de tamaño pequeño
Este modo muestra las etiquetas iniciales y finales de los elementos y atributos sin el nombre del elemento o atributo:

 © 2015 Altova GmbH


Nanonull, Inc.
Location: US
Street: 119 Oakstreet, Suite 4876
City: Vereno
State & Zip: DC 29213
Phone: +1 (321) 555 5155 0
Fax: +1 (321) 555 5155 4
E-mail: office@nanonull.com
Vereno Office Summary: 4 departments, 15 employees.
The company was established in Vereno in 1995 as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed NanoSoft Development Suite in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.

Observe que las etiquetas iniciales incluyen un símbolo, mientras que las etiquetas finales están vacías. Asimismo, las etiquetas de elementos llevan un símbolo en forma de corchetes angulares, mientras que las etiquetas de atributos llevan el signo = (*ver imagen siguiente*).

Para contraer o expandir un elemento o atributo, haga doble clic en su etiqueta. El ejemplo que aparece a continuación muestra un elemento contraído (resaltado en color azul). Observe la forma de la etiqueta del elemento contraído, comparada con la etiqueta inicial del elemento expandido situada a la izquierda.

Marcado mixto
El modo de marcado mixto muestra el marcado según las opciones de configuración elegidas por la persona que diseñó la StyleVision Power Stylesheet (por cada elemento y atributo del documento puede indicar si el elemento o atributo debe mostrarse con marcado grande, marcado pequeño o sin marcado). El usuario de la vista Authentic ve este marcado personalizado en el modo de marcado mixto.

Ocultar marcado
Se oculta el marcado XML. Puesto que el formato visible en la vista Authentic es el formato del documento impreso, este modo de visualización es una vista WYSIWYG del documento.

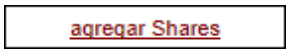
Visualización de contenidos
En la vista Authentic el contenido se puede visualizar de dos maneras:

- Texto sin formato. El usuario escribe el texto y este texto se convierte en el contenido del elemento o en el valor del atributo.

- Mecanismos de entrada de datos. La presentación incluye un campo de entrada de datos (un cuadro de texto), un campo de entrada multilínea, un cuadro combinado, una casilla de verificación o un botón de opción. En el caso de campos de entrada y campos de entrada multilínea, el texto que se escriba en el campo se convierte en el contenido XML del elemento o en el valor del atributo.

  En el caso de los demás mecanismos de entrada de datos, la opción seleccionada produce un valor XML equivalente, que viene especificado en la StyleVision Power Stylesheet. Por tanto, si en un cuadro combinado selecciona, por ejemplo, la opción "confirmado" (que estaría disponible en la lista desplegable del cuadro combinado) al elemento se le asignaría el valor XML "1" o el valor "confirmado", por ejemplo. Si la opción seleccionada fuese "no confirmado", se le asignaría el valor XML "0" o el valor "no confirmado", por ejemplo.

Nodos opcionales
Cuando un elemento o atributo es opcional (de acuerdo con el esquema al que se hace referencia), aparece un aviso de tipo `agregar [nombre del elemento/atributo]`:

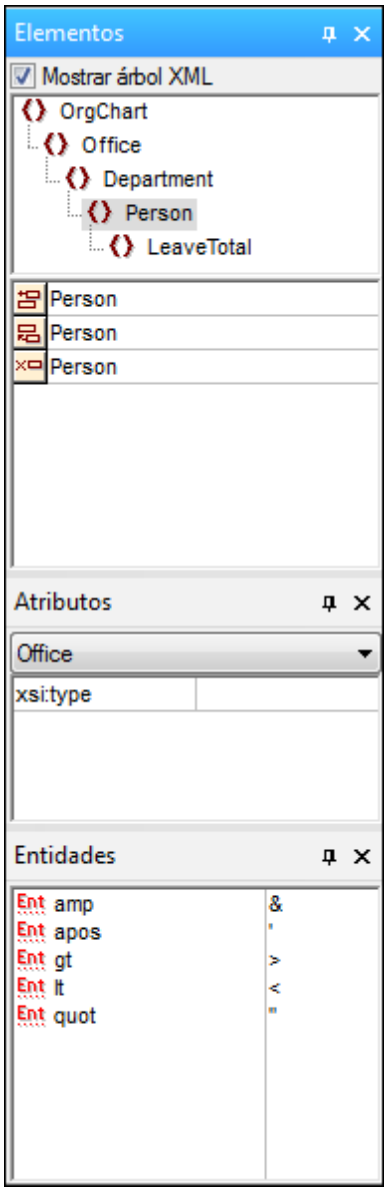


Al hacer clic en el aviso se agrega el elemento y el cursor se coloca en el campo de entrada de datos. Si hay varios nodos opcionales, aparece el aviso `agregar...` . Al hacer clic en el aviso, aparece un menú con los nodos opcionales.

© 2015 Altova GmbH

3.4 Ayudantes de entrada de la vista Authentic

La vista Authentic ofrece tres ayudantes de entrada: Elementos, Atributos y Entidades. Los ayudantes de entrada están disponibles en tres ventanas situadas a la derecha de la interfaz de la vista Authentic (*imagen siguiente*).



Los ayudantes de entrada Elementos y Atributos son contextuales, es decir, sus opciones dependen de la posición del cursor en el documento. Las entidades del ayudante de entrada Entidades no son contextuales, es decir, el ayudante muestra todas las entidades permitidas en el documento, independientemente de la posición del cursor.

A continuación describimos los ayudantes de entrada uno por uno.

Ayudante de entrada Elementos

El ayudante de entrada Elementos está dividido en dos paneles:

- Un panel superior con el árbol XML (que se puede activar o desactivar haciendo clic en la casilla **Mostrar árbol XML**). El árbol XML muestra los antecesores del elemento seleccionado hasta la altura del elemento raíz del documento. Al hacer clic en un elemento del árbol XML, los elementos relevantes para ese elemento (que se describen en el siguiente punto) aparecen en el panel inferior del ayudante de entrada Elementos.
- Un panel inferior con los nodos que se pueden eliminar, insertar en la posición actual, insertar antes de la posición actual e insertar después de la posición actual. La lista del panel inferior también incluye los nodos que se pueden aplicar al elemento o intervalo de texto seleccionado y los que se pueden borrar del elemento o intervalo de texto en la vista Authentic. Los nodos de esta lista van acompañados por un icono que indica el tipo de acción que se puede realizar. A continuación describimos cada uno de estos iconos y su significado.

Para utilizar un nodo del ayudante de entrada Elementos, haga clic en su icono.

Insertar después del elemento
El elemento del ayudante de entrada se inserta después del elemento seleccionado y se anexa en el nivel jerárquico correcto. Por ejemplo, si el cursor está dentro de un elemento `//sect1/para` y anexa un elemento `sect1`, el elemento `sect1` nuevo no se anexará como elemento del mismo nivel que `//sect1/para` sino como elemento del mismo nivel que `sect1`, que es el elemento primario del elemento `para`.

Insertar antes del elemento
El elemento del ayudante de entrada se inserta antes del elemento seleccionado. Tenga en cuenta que, al igual que el comando **Insertar después del elemento**, el elemento se inserta en el nivel jerárquico correcto.

Quitar elemento
Quita el elemento y su contenido.

Insertar elemento
También puede insertar un elemento del ayudante de entrada dentro de otro elemento. Cuando el cursor está dentro de un elemento, entonces se pueden insertar los elementos secundarios permitidos de ese elemento. Recuerde que los elementos secundarios permitidos pueden formar parte de un modelo de contenido que sólo contiene elementos o de un modelo de contenido mixto (con elementos y texto).

Para insertar un elemento secundario permitido, seleccione un intervalo de texto o sitúe el cursor en una posición dentro del texto (el punto de inserción).

- Cuando se selecciona un intervalo de texto y se inserta un elemento, el intervalo de texto se convierte en el contenido del elemento insertado.
- Cuando se inserta un elemento en un punto de inserción, se inserta el elemento en ese punto.

Después de insertar el elemento, puede borrarlo con solo hacer clic en el icono **Borrar elemento** del ayudante de entrada Elementos, disponible para estos elementos insertados. El icono **Borrar elemento** puede ser de dos tipos, dependiendo de si selecciona un intervalo de texto o sitúa el cursor en el texto

 © 2015 Altova GmbH

para indicar un punto de inserción (véase la descripción que aparece más abajo).

Aplicar elemento
Si selecciona un elemento del documento (active el modo de vista "Mostrar marcado en tamaño grande" y seleccione la etiqueta inicial o final del elemento) y ese elemento se puede sustituir por otro elemento (p. ej. en un elemento de contenido mixto como `para`, el elemento `italic` se puede sustituir por el elemento `bold`), este icono indica que el elemento del ayudante de entrada se puede aplicar al elemento seleccionado (el elemento original). El comando **Aplicar elemento** también se puede aplicar a un intervalo de texto de un elemento de contenido mixto. El intervalo de texto se convertirá en el contenido del elemento que se aplicó.

- Si el elemento que se aplicó tiene **un secundario con el mismo nombre** que el secundario del elemento original y existe una instancia de este secundario en el elemento original, entonces se mantiene el secundario del elemento original en el contenido del elemento nuevo.
- Si el elemento que se aplicó **no tiene un secundario con el mismo nombre** que el secundario del elemento original y existe una instancia del secundario en el elemento original, entonces el secundario del elemento original se anexa como elemento del mismo nivel de los secundarios del elemento nuevo.
- Si el elemento que se aplicó tiene **un secundario para el que no hay un equivalente** en el modelo de contenido del elemento original, entonces este elemento secundario no se crea automáticamente, aunque la vista Authentic le da la opción de insertarlo.

Si en lugar de seleccionar un elemento, selecciona un intervalo de texto y pulsa el icono **Aplicar elemento**, entonces se crea el elemento del ayudante de entrada en esa posición y el texto seleccionado se convierte en el contenido del elemento recién creado. No se puede aplicar un elemento cuando el cursor está situado en un punto de inserción.

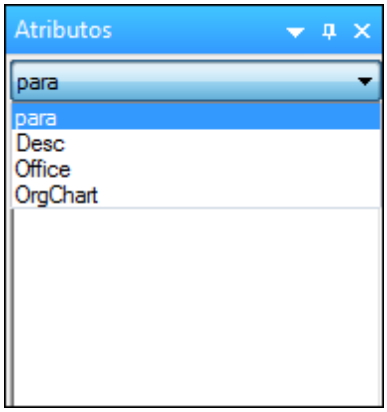
Borrar elemento (cuando se selecciona un texto)
Este icono aparece cuando se selecciona texto dentro de un elemento de contenido mixto. Si hace clic en el icono, se borra el elemento que rodea el texto seleccionado.

Borrar elemento (cuando se selecciona un punto de inserción)
Este icono aparece cuando se sitúa el cursor dentro de un elemento que es un secundario de un elemento de contenido mixto. Si hace clic en el icono, se borra el elemento alineado.

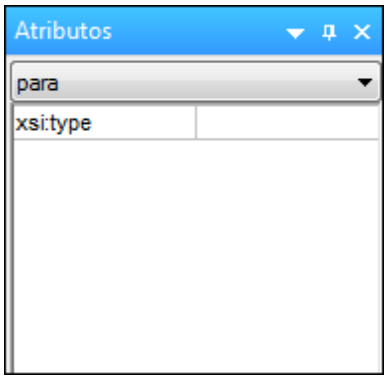
Ayudante de entrada Atributos
El ayudante de entrada Atributos contiene un cuadro combinado con una lista desplegable. Dentro del cuadro combinado aparece el elemento seleccionado (seleccione la etiqueta inicial o final de un elemento o sitúe el cursor dentro del contenido del elemento).

El ayudante de entrada que aparece en estas imágenes incluye un elemento `para` en el cuadro combinado. Al hacer clic en la punta de flecha del cuadro combinado, se despliega una lista con todos los ancestros del elemento `para` **hasta llegar el elemento raíz del documento**, en este

caso `OrgChart`.



Justo debajo del cuadro combinado aparece una lista de atributos válidos para el elemento seleccionado en el cuadro combinado, en este caso el elemento `para`. Si un atributo es obligatorio en un elemento dado, el atributo aparece en negrita. (En la imagen siguiente, no hay atributos obligatorios, excepto el atributo integrado `xsi:type`.)



Para introducir el valor de un atributo, haga clic en el campo de valor del atributo y escriba el valor. Como resultado se crea el atributo y su valor en el documento XML.

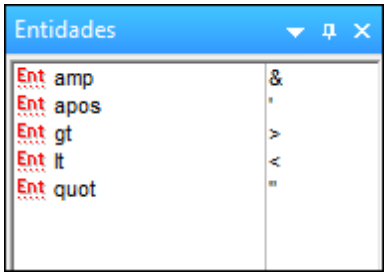
En el caso del atributo `xsi:nil`, que aparece en el ayudante de entrada Atributos cuando se selecciona un elemento que puede tener valores nulos, entonces el valor del atributo `xsi:nil` sólo se puede introducir seleccionando los valores permitidos (`true` o `false`) de la lista desplegable del campo de valor del atributo.

Para cambiar el atributo `xsi:type` haga clic en el campo de valor del atributo y, en la lista desplegable que aparece, haga clic en uno de los valores. Los valores de la lista son los tipos abstractos disponibles definidos en el esquema XML en el que se basa el documento de la vista Authentic.

Ayudante de entrada Entidades

El ayudante de entrada Entidades permite insertar entidades en el documento. Las entidades se pueden usar para insertar caracteres especiales o fragmentos de texto que aparecen con frecuencia en el documento (como, por ejemplo, el nombre de una compañía). Para insertar una entidad, sitúe el cursor en la posición donde desea insertar la entidad y haga doble clic en la entidad en el ayudante de entrada Entidades.

© 2015 Altova GmbH



Nota: se denomina *entidad interna* aquella entidad cuyo valor está definido dentro de una DTD. Por el contrario, *entidad externa* es aquella cuyo valor está en una fuente externa, p. ej. otro archivo XML. Tanto las entidades externas como las internas aparecen en el ayudante de entrada Entidades. Cuando se inserta una entidad, ya sea externa o interna, se inserta la entidad en el texto XML, no sólo su valor. Si la entidad es interna, la vista Authentic **muestra el valor de la entidad**. Si la entidad es externa, la vista Authentic muestra la entidad, pero no su valor. Esto significa que, por ejemplo, en la vista Authentic un archivo XML que sea una entidad externa se mostrará como entidad. Su contenido no sustituirá a la entidad en la vista Authentic.

En la vista Authentic también puede definir entidades nuevas, que aparecerán también en el ayudante de entrada. Para más información consulte el apartado Definir entidades de la sección *Edición en la vista Authentic*.

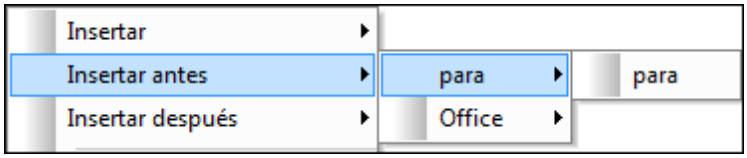
3.5 Menús contextuales de la vista Authentic

Al hacer clic con el botón secundario en un nodo del documento o en el contenido, aparece un menú contextual con comandos relacionados con la selección o la posición del cursor.

Insertar elementos

La imagen siguiente muestra el submenú **Insertar**, que incluye una lista con todos los elementos que se pueden insertar en la posición del cursor. El submenú **Insertar antes** ofrece una lista con todos los elementos que se pueden insertar antes del elemento seleccionado. El submenú **Insertar después** ofrece una lista con todos los elementos que se pueden insertar después del elemento seleccionado. En la imagen siguiente, el elemento seleccionado es `para` y dentro del elemento `para` se pueden insertar los elementos `ts:bold`, `ts:italic` y `ts:underline`.

Como muestra la imagen siguiente, antes del elemento `para` se pueden insertar los elementos `para` y `Office`.



Los comandos de inserción de nodo, sustitución (**Aplicar**) y eliminación de marcado (**Quitar**) del menú contextual también están disponibles en los ayudantes de entrada de la vista Authentic y se describen en detalle en esa sección.

Insertar entidad

Al pasar el puntero sobre el comando **Insertar entidad** aparece un submenú con una lista de todas las entidades declaradas. Al hacer clic en una entidad de la lista, la entidad se inserta en la posición seleccionada. Para más información sobre cómo definir entidades para sus documentos, consulte la sección Definir entidades.

Insertar sección CDATA

Este comando se habilita cuando se sitúa el cursor dentro de un texto. Al hacer clic en este comando, se inserta una sección CDATA en el punto de inserción del cursor. La sección CDATA está delimitada por una etiqueta inicial y otra final. Para ver las etiquetas, active el marcado en tamaño grande o tamaño pequeño. Dentro de las secciones CDATA se ignora el marcado XML y no se analizan caracteres. Los caracteres de marcado XML (como & ' > < ") no se analizan como marcado, sino como literales. Por tanto, las secciones CDATA son ideales a la hora de usar código de programa de muestra con caracteres XML en el documento.

Quitar nodo

Al pasar el puntero sobre el comando **Quitar** aparece un submenú con una lista que incluye el nodo seleccionado y todos sus ancestros que se puedan quitar (si al quitar estos ancestros el

 © 2015 Altova GmbH

documento sigue siendo válido). Haga clic en el elemento que desea quitar. Tenga en cuenta que si hace clic en un ancestro, se eliminará el elemento y sus descendientes, incluido el elemento seleccionado.

Borrar

El comando **Borrar** elimina el marcado del elemento que envuelve la selección. Si se selecciona todo el nodo, entonces se elimina el marcado de todo el nodo. Si se selecciona un fragmento de texto, se elimina el marcado que envuelve sólo el fragmento seleccionado.

Aplicar

El comando **Aplicar** aplica un elemento seleccionado al nodo o texto seleccionado en la ventana principal. Para más detalles, consulte el apartado ayudantes de entrada de la vista Authentic.

Copiar, cortar y pegar

Son los comandos estándar de Windows. No obstante, tenga en cuenta que el comando **Pegar** pega el texto copiado bien como XML, bien como texto, dependiendo de lo especificado en la hoja de estilos SPS. Para más información sobre los comandos **Copiar como XML** y **Copiar como texto**, consulte la descripción del párrafo siguiente sobre el comando **Copiar como**.

Pegar como

El comando Pegar como ofrece la opción de pegar un fragmento XML de la vista Authentic (que se copió en el portapapeles) como XML o como texto. Si el fragmento copiado se pega como XML, se pega e incluye el marcado XML. Si se pega como texto, entonces se pega sólo el contenido de texto del fragmento copiado (sin el marcado XML). Se pueden dar estos dos casos:

- Se selecciona ***todo un nodo y sus etiquetas de marcado*** en la vista Authentic y se copia en el portapapeles. (i) El nodo se puede pegar como XML en cualquier posición, siempre que la posición sea válida. No se puede pegar en una posición no válida. (ii) Si el nodo se pega como texto, se pega sólo el *contenido de texto* del nodo (no su marcado). El texto se puede pegar en cualquier posición del documento XML que acepte texto.
- Se selecciona un ***fragmento de texto*** en la vista Authentic y se copia en el portapapeles. (i) Si este fragmento se pega como XML, las etiquetas de marcado XML del texto se pegan con el texto (aunque no se copiaran), pero sólo si el nodo XML es válido en la posición donde se pega el fragmento. (ii) Si el fragmento se pega como texto, entonces se puede pegar en cualquier posición del documento XML que acepte texto.

Nota: El texto se copiará en nodos que acepten texto y el usuario debería comprobar si, tras copiar el texto, el documento sigue siendo válido. Recuerde que el texto copiado:

(i) debería ser válido desde un punto de vista léxico en la posición nueva (por ejemplo, no sería válido pegar caracteres numéricos en un nodo de tipo numérico) y
(ii) no debería invalidar el nodo en la posición nueva (por ejemplo, no sería válido pegar cuatro dígitos en un nodo que acepta números con un máximo de tres dígitos).

Si el texto pegado invalida el documento, el texto aparecerá en color rojo a modo de aviso.

Eliminar

El comando **Eliminar** elimina el nodo seleccionado y su contenido. Para seleccionar

el nodo que desea eliminar, sitúe el cursor dentro del nodo o haga clic en su etiqueta inicial o final.

© 2015 Altova GmbH

4 Edición en la vista Authentic

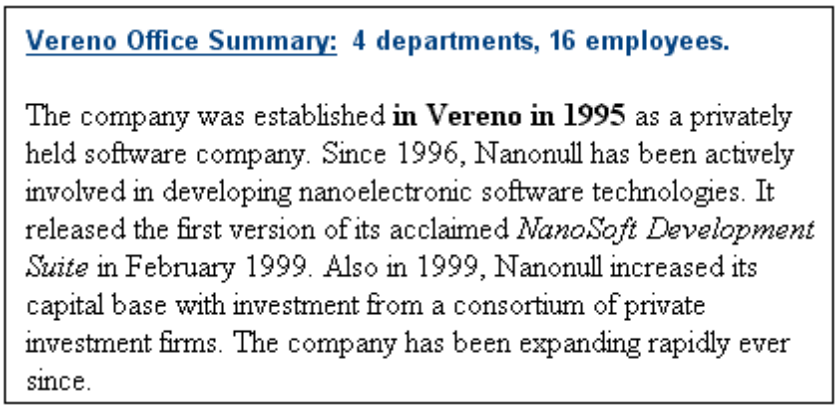
En esta sección describimos con detalle algunas funciones de la vista Authentic que consideramos importantes. Estas funciones son importantes bien porque se utilizan con mucha frecuencia o bien porque implican una cierta complejidad y deben ser explicadas en más detalle.

Los temas que describimos en esta sección son:

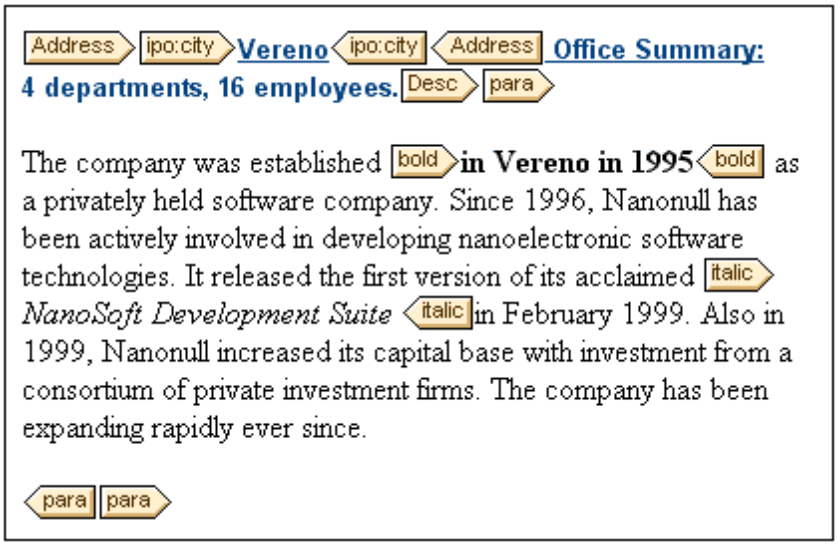
- En la vista Authentic se pueden usar tres tipos de tablas diferentes. El apartado Tablas en la vista Authentic describe los tres tipos de tablas (tablas SPS estáticas, tablas SPS dinámicas y tablas XML) así como cuándo y cómo se utilizan. Este apartado empieza con una descripción general y después explica detalladamente el uso de las tablas.
- El selector de fechas es un calendario gráfico con el que puede introducir fechas en el formato XML correcto con solo hacer clic en una fecha. Consulte el apartado titulado Selector de fecha.
- Una entidad es una abreviación de un carácter especial o una cadena de texto. En la vista Authentic puede definir entidades propias para poder insertar estos caracteres o cadenas de texto especiales. Para más información consulte el apartado Definir entidades.
- Qué formatos de imagen pueden visualizarse en la vista Authentic.

4.1 Edición básica

En la vista Authentic puede editar documentos XML. Sin embargo, la vista Authentic puede ocultar el marcado XML del documento y mostrar solamente el contenido del documento (*ver la primera imagen de este apartado*). De este modo, el usuario no entra en contacto con la tecnología XML y puede editar el documento como si se tratase de un documento de texto normal y corriente. No obstante, si lo prefiere, puede activar el marcado XML del documento en cualquier momento (*ver la segunda imagen de este apartado*).


Vereno Office Summary: 4 departments, 16 employees.
The company was established in Vereno in 1995 as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed NanoSoft Development Suite in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.

Un documento editable en la vista Authentic sin marcado XML.


Address
ipo:city
Vereno
ipo:city
Address
Office Summary:
4 departments, 16 employees.
Desc
para
The company was established
bold
in Vereno in 1995
bold
as a privately held software company. Since 1996, Nanonull has been actively involved in developing nanoelectronic software technologies. It released the first version of its acclaimed
italic
NanoSoft Development Suite
italic
in February 1999. Also in 1999, Nanonull increased its capital base with investment from a consortium of private investment firms. The company has been expanding rapidly ever since.
para
para

Un documento editable en la vista Authentic con las etiquetas de marcado XML.

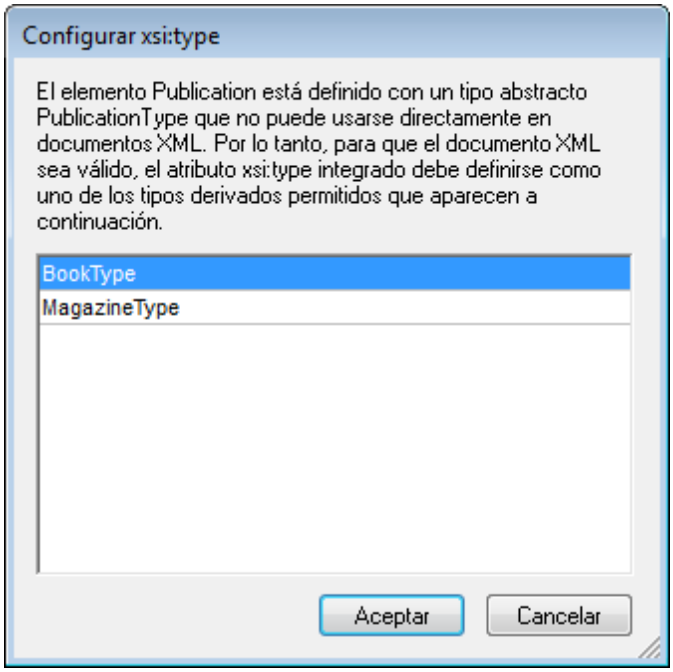
Insertar nodos

A menudo es necesario agregar nodos nuevos al documento XML de Authentic. Por ejemplo, puede que necesite agregar un elemento `Person` nuevo en una libreta de direcciones. En estos casos el esquema XML permitiría añadir el elemento nuevo. Usted sólo tiene que hacer clic con el botón secundario en el nodo del documento antes o después del cual desea agregar el nodo nuevo. En el menú contextual que aparece seleccione **Insertar antes** o **Insertar después** según corresponda. Los nodos que se pueden insertar en la posición elegida aparecen en un submenú. Haga

 © 2015 Altova GmbH

clic en el nodo que desea insertar y este se insertará en la posición elegida. Todos los nodos descendientes obligatorios se insertarán también. Si un nodo descendiente es opcional, entonces aparece el hipervínculo `Agregar nombreDelNodo` para que pueda insertar el nodo si lo desea.

Si el nodo que se añade es un elemento con un tipo abstracto, entonces aparece un cuadro de diálogo (*similar al de la imagen siguiente*) con una lista de los tipos derivados que están disponibles en el esquema XML.


Configurar xsi:type
El elemento Publication está definido con un tipo abstracto PublicationType que no puede usarse directamente en documentos XML. Por lo tanto, para que el documento XML sea válido, el atributo xsi:type integrado debe definirse como uno de los tipos derivados permitidos que aparecen a continuación.
BookType
MagazineType
Aceptar
Cancelar

El diálogo anterior aparece cuando se añade un elemento `Publication`. El elemento `Publication` es de tipo `PublicationType`, que es un tipo complejo abstracto. Los dos tipos complejos `BookType` y `MagazineType` se derivan del tipo complejo abstracto `PublicationType`. Por tanto, cuando se añade un elemento `Publication` al documento XML, es necesario especificar uno de estos dos tipos concretos derivados del tipo abstracto de `Publication`. El nuevo elemento `Publication` se añadirá con un atributo `xsi:type`:

```
<Publication xsi:type="BookType"> ... </Publication>
<Publication xsi:type="MagazineType"> ... </Publication>
<Publication xsi:type="MagazineType"> ... </Publication>
```

Al seleccionar uno de los tipos derivados disponibles y hacer clic en **Aceptar** ocurre esto:

- el tipo derivado seleccionado se establece como valor del atributo `xsi:type` del elemento nuevo y
- se inserta el elemento junto con sus nodos descendientes definidos en el modelo de contenido del tipo derivado seleccionado.

El tipo derivado seleccionado se puede cambiar más tarde. Para ello cambie el valor del atributo `xsi:type` del elemento en el ayudante de entrada Atributos. Cuando cambie el tipo del elemento de esta forma, se quitarán del documento todos los nodos del modelo de contenido del tipo anterior y se insertarán los nodos del modelo de contenido del tipo nuevo.

Edición de texto

Los documentos de la vista Authentic están compuestos por texto e imágenes. Para editar el texto del documento, sitúe el cursor en la posición donde desea insertar texto y empiece a teclear. Puede copiar, mover y eliminar texto de forma normal (por ejemplo con la tecla **Suprimir**) y mediante operaciones de arrastrar y colocar. La única excepción es la tecla **Entrar**. Puesto que el documento de la vista Authentic viene con formato previo, el usuario no puede agregar líneas ni espacios entre los componentes. Por tanto, en la vista Authentic la tecla **Entrar** sirve para anexar otra instancia del elemento que se está editando y sólo se puede usar para esto.

Copiar como XML o como texto

Puede copiar y pegar texto como XML o como texto.

- Si el texto se pega como XML, entonces se pega el marcado XML y el contenido de texto de los nodos. El marcado XML se pega incluso si solo copió parte del contenido del nodo. Para poder pegar el marcado es necesario que el esquema permite insertar el nodo en la posición elegida.
- Si el texto se pega como texto, entonces no se pega el marcado XML.

Para pegar el texto como XML o como texto, antes debe copiar el texto (**Ctrl+C**), hacer clic con el botón secundario en la posición donde desea pegarlo y seleccionar **Pegar como | XML** o **Pegar como | Texto** en el menú contextual. Si utiliza **Ctrl+V** para pegar el texto, este se pega en el modo predeterminado de la hoja de estilos SPS. El modo predeterminado lo especifica el diseñador de la hoja de estilos SPS. Para más información, consulte el apartado Menús contextuales.

También puede resaltar un texto y arrastrarlo hasta la posición donde desea pegarlo. Al soltar el texto en su nueva posición aparece un mensaje emergente preguntando si desea pegarlo como texto o como XML.

Formato de texto

Un principio fundamental de los documentos XML es que el contenido se puede separar de su presentación. El documento XML lleva el contenido, mientras que la hoja de estilos contiene la presentación (el formato). En la vista Authentic el documento XML se presenta por medio de la hoja de estilos. Esto significa que todo el formato que ve en la vista Authentic lo produce la hoja de estilos. Si ve texto en negrita, esta opción de formato viene dada por la hoja de estilos. Si ve una lista o una tabla, esa opción de formato también viene dada por la hoja de estilos. El documento XML, que usted edita en la vista Authentic, contiene únicamente el contenido, pero no formato. El formato está definido en la hoja de estilos. Esto significa que en la vista Authentic el usuario no puede aplicar formato al texto que está editando, porque está editando contenido. El formato que se aplica automáticamente al contenido que está editando está conectado al valor semántico o estructural de los datos que está editando. Por ejemplo, a una dirección de correo electrónico (que puede entenderse como una unidad semántica) se le aplica determinado formato porque se trata de una dirección de correo electrónico. Asimismo, un título debe aparecer en una posición determinada del documento (es una unidad semántica y estructural) y se le aplica determinado formato en base a lo indicado por el diseñador de la hoja de estilos. Por tanto, el usuario no puede cambiar el formato ni de las direcciones de correo electrónico ni de los títulos. Lo único que puede hacer es editar el contenido de las direcciones de correo electrónico y de los títulos.

En algunos casos es necesario presentar el contenido de manera especial. Por ejemplo, quizás sea necesario presentar cierta cadena de texto en negrita. En ese caso, la presentación debe estar vinculada a un elemento estructural del documento. Por ejemplo, el diseñador de la hoja de

© 2015 Altova GmbH

estilos separa estructuralmente la cadena de texto del resto de contenido. Si usted necesita usar dicha cadena de texto en la vista Authentic, debe introducir la cadena de texto dentro del marcado adecuado. Para más información, consulte el comando **Insertar elemento** del apartado Ayudante de entrada Elementos.

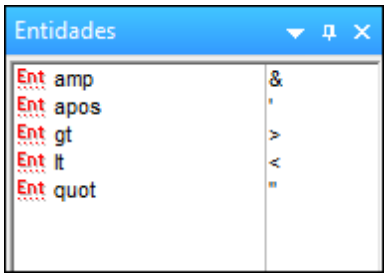
Usar RichEdit en la vista Authentic

En la vista Authentic los botones y controles de la barra de herramientas RichEdit (*imagen siguiente*) se habilitan cuando se coloca el cursor dentro de un elemento que se creó como componente RichEdit. De lo contrario, las opciones de esta barra de herramientas aparecen atenuadas.

Seleccione el texto que desea editar y después utilice los botones y controles de la barra de herramientas RichEdit para seleccionar el estilo que desea aplicar al texto. RichEdit permite al usuario de la vista Authentic especificar la fuente, su peso, estilo, decoración, tamaño, color, color de fondo y alineación. El texto se incluirá entre las etiquetas que corresponden a los estilos elegidos.

Insertar entidades

En los documentos XML algunos caracteres no se pueden utilizar como texto normal porque son caracteres XML utilizados para el marcado. Se trata de los caracteres et (`&`), apóstrofe (`'`), menor que (`<`), mayor que (`>`) y comillas inglesas (`"`). Si desea usar estos caracteres en el texto del documento, debe insertarlos como referencias de entidad con ayuda del ayudante de entrada Entidades (*imagen siguiente*).



En XML también puede crear sus propias entidades. Puede crear entidades para (i) caracteres especiales que no están disponibles en su teclado, (ii) cadenas de texto que desea utilizar con frecuencia en su documento, (iii) fragmentos de código XML o (iv) imágenes y otros recursos. Estas entidades se pueden definir en la vista Authentic directamente. Después de definirlas, estas entidades aparecen en el ayudante de entrada Entidades y se pueden insertar en el documento.

Insertar secciones CDATA

Las secciones CDATA son secciones de texto de un documento XML que el analizador XML no procesa como datos XML. Pueden utilizarse para escapar secciones de texto grande, como fragmentos de código de programa o XML que el usuario desea reproducir con sus etiquetas de marcado. Las secciones CDATA pueden aparecer dentro del contenido de un elemento y están delimitadas por `<![CDATA[` y `]]>`. Esto significa que la cadena de texto `]]>` no se debe utilizar dentro de una sección CDATA puesto que ello indicaría el fin de la sección. Si desea utilizar la

cadena `]]>` dentro de la sección CDATA, entonces debe sustituir el carácter `>` por su referencia de entidad (`>`). Para insertar una sección CDATA dentro de un elemento, haga clic con el botón secundario en la posición elegida y seleccione **Insertar sección CDATA** en el menú contextual. Para ver las etiquetas de la sección CDATA en la vista Authentic, active la visualización del marcado. Otra manera de crear una sección CDATA es resaltar el texto y después seleccionar el comando **Insertar sección CDATA**.

Nota: las secciones CDATA no se pueden insertar en campos de entrada (es decir, en cuadros de texto y cuadros de texto multilínea). Las secciones CDATA sólo se pueden introducir en elementos que aparecen como componentes de texto en la vista Authentic.

Editar y usar hipervínculos

Un hipervínculo está formado por el texto del vínculo y el destino del vínculo. Para editar el texto del hipervínculo haga clic en el texto y edítelo. El destino del hipervínculo no se puede editar. (El destino del hipervínculo lo define el diseñador de la hoja de estilos, escribiendo la dirección de destino estática o derivando la dirección de destino de los datos que contiene el documento XML). Para ir al destino del hipervínculo desde la vista Authentic haga clic en el texto del hipervínculo mientras pulsa la tecla **Ctrl**. (Recuerde que si solo hace clic en el hipervínculo entonces se habilita la edición del texto del hipervínculo.)

© 2015 Altova GmbH

4.2 Tablas en la vista Authentic

Hay tres tipos de tablas, que a su vez se agrupan en dos categorías: tablas SPS (estáticas y dinámicas) y tablas CALS/HTML.

Hay dos tipos de **tablas SPS**: estáticas y dinámicas. Las tablas SPS las diseña el diseñador de la hoja de estilos SPS a la que está vinculado el documento XML. Usted no puede insertar una tabla SPS en el documento XML, pero puede introducir datos en los campos de las tablas SPS y agregar y eliminar filas en las tablas SPS dinámicas. El apartado tablas SPS que aparece a continuación explica las características de este tipo de tablas.

Lo que sí puede insertar el usuario de la vista Authentic son **tablas CALS/HTML**. El propósito de estas tablas es que el usuario pueda insertar tablas en cualquier posición de la jerarquía del documento. Las características de edición de las tablas CALS/HTML y de los iconos de edición para tablas CALS/HTML se describen en los apartados siguientes.

4.2.1 Tablas SPS

En la vista Authentic se pueden usar dos tipos de tablas SPS: estáticas y dinámicas.

Las **tablas estáticas** tienen una estructura y un tipo de contenido fijos. El usuario de la vista Authentic puede introducir datos en las celdas de la tabla pero no puede cambiar su estructura (es decir, no puede agregar filas ni columnas, etc.) ni cambiar el tipo de contenido de las celdas. Puede introducir los datos a mano o seleccionando una de las opciones disponibles en forma de casilla de verificación, botón de opción o lista desplegable de cuadros combinados.

Nanonull, Inc.

Street:	119 Oakstreet, Suite 4876	**Phone:**	+1 (321) 555 5155
City:	Vereno	**Fax:**	+1 (321) 555 5155 - 9
State & Zip:	DC 29213	**E-mail:**	office@nanonull.com

Nota importante: los iconos y comandos para editar tablas dinámicas **no se deben usar** para editar tablas estáticas.

Las **tablas dinámicas** tienen filas que representan una estructura de datos repetitiva, es decir, cada fila tiene una estructura de datos idéntica (al contrario de las tablas estáticas). Por tanto, en este tipo de tablas puede anexar, insertar, subir, bajar y eliminar filas. Estos comandos están disponibles en el menú **Authentic**, así como en la barra de herramientas (*ver iconos de la imagen*).

Para realizar estas operaciones, ponga el cursor en cualquier posición dentro de la fila que desea editar y después seleccione el comando pertinente.

Administration (3)

First	Last	Title	Ext	EMail	Shares	Leave		
						Total	Used	Left
Vernon	**Callaby**	Office Manager	582	v.callaby@nanonull.com	1500	25	4	21
Frank	Further	Accounts Receivable	471	f.further@nanonull.com	0	22	2	20
Loby	Matise	Accounting Manager	963	l.matise@nanonull.com	agregar Shares	25	7	18
Employees: 3 (20% of Office, 8% of Company)					Shares: 1500 (12% of Office, 4% of Company)			
Non-Shareholders: Frank Further, Loby Matise.								

Para desplazarse por las celdas de la tabla, use las teclas de dirección. Para avanzar de una celda a la siguiente, use la tecla **Tabulador**. Si el cursor está en la última celda de la última fila y pulsa la tecla **Tabulador**, se crea una fila nueva.

© 2015 Altova GmbH

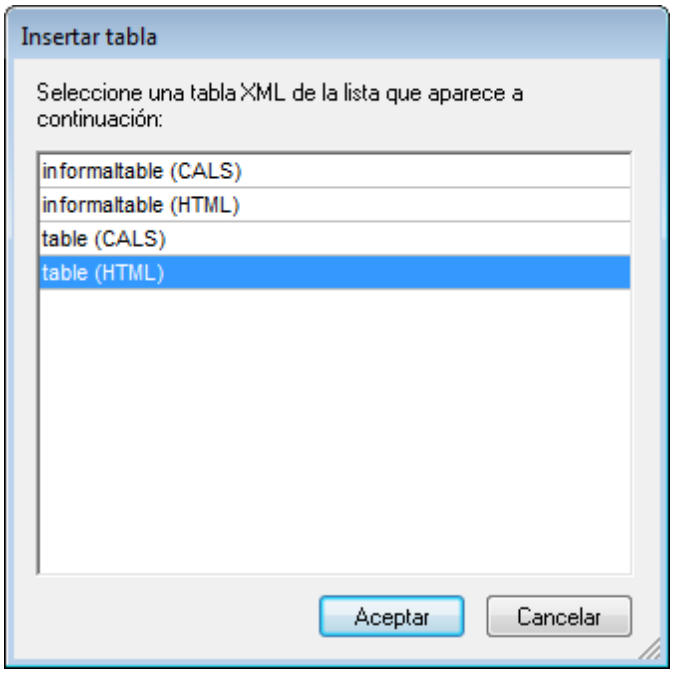
4.2.2 Tablas CALS/HTML

El usuario de la vista Authentic puede insertar tablas CALS/HTML para aquellas estructuras de datos XML que se diseñaran para ser presentadas en formato de tabla. El trabajo con tablas CALS/HTML implica tres pasos: insertar la tabla, darle formato e introducir los datos. Los comandos para trabajar con tablas CALS/HTML cuentan con iconos en la barra de herramientas (véase Iconos de edición para tablas CALS/HTML).

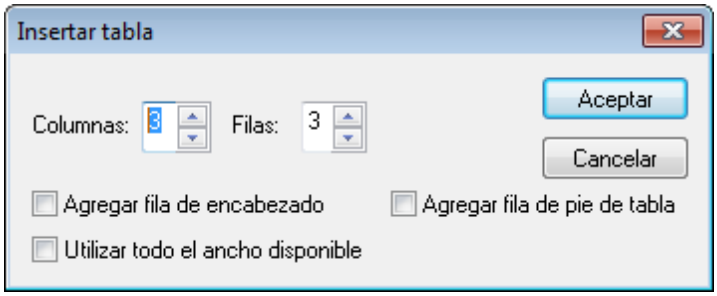
Insertar tablas

Para insertar una tabla CALS/HTML siga estos pasos:

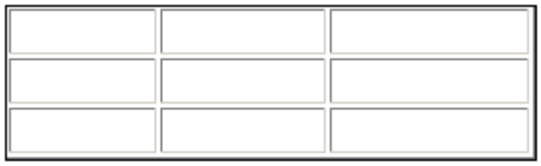
1. Ponga el cursor en la posición donde desea insertar la tabla y haga clic en el icono . (Recuerde que es el esquema el que determina dónde está permitido insertar una tabla). Aparece el cuadro de diálogo "Insertar tabla" (*imagen siguiente*). Este diálogo enumera todas las estructuras de datos de elementos XML para las que se definieron estructuras de tabla. En el ejemplo de la imagen que aparece a continuación puede ver que los elementos `table` y `informaltable` se definieron como tablas CALS y HTML.



2. Seleccione el elemento y el tipo de tabla que desea insertar y haga clic en **Aceptar**.
3. Aparece otro diálogo (*imagen siguiente*). Seleccione el número de columnas y filas para la tabla, indique si desea añadir un encabezado o pie de tabla y si la tabla debe ocupar todo el ancho de la página. Después haga clic en **Aceptar**.



Con la configuración de la imagen anterior se obtendría esta tabla:



Con los comandos del menú **Tabla** puede agregar y eliminar columnas, así como combinar y dividir filas y columnas. Pero para empezar debe crear la estructura general de la tabla.

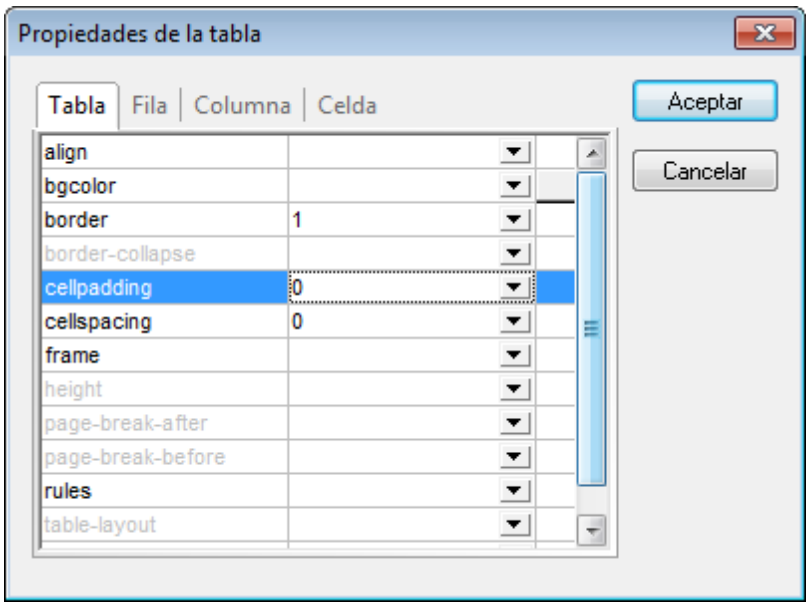
Cómo dar formato a las tablas e introducir datos

En el diseño del documento ya se asignó un formato a la tabla. No obstante, puede modificar el formato de la tabla en algunas circunstancias:

- Los elementos que se corresponden a los elementos de estructura de la tabla deben tener definidos como atributos (en el esquema XML subyacente) las propiedades de tabla CALS o HTML correspondientes. Para el formato de la tabla sólo se pueden usar los atributos que están definidos en el esquema XML. Si en el diseño se establecieron valores para estos atributos, entonces puede cambiar los valores de estos atributos en la vista Authentic.
- En el diseño el atributo `style` no puede contener estilos CSS. Si se definió un atributo `style` con estilos CSS para un elemento, entonces el atributo `style` tiene prioridad sobre el resto de los atributos de formato del elemento. Como resultado, se reemplazan las opciones de formato indicadas en la vista Authentic.

Siga estos pasos para dar formato a una tabla, fila, columna o celda:

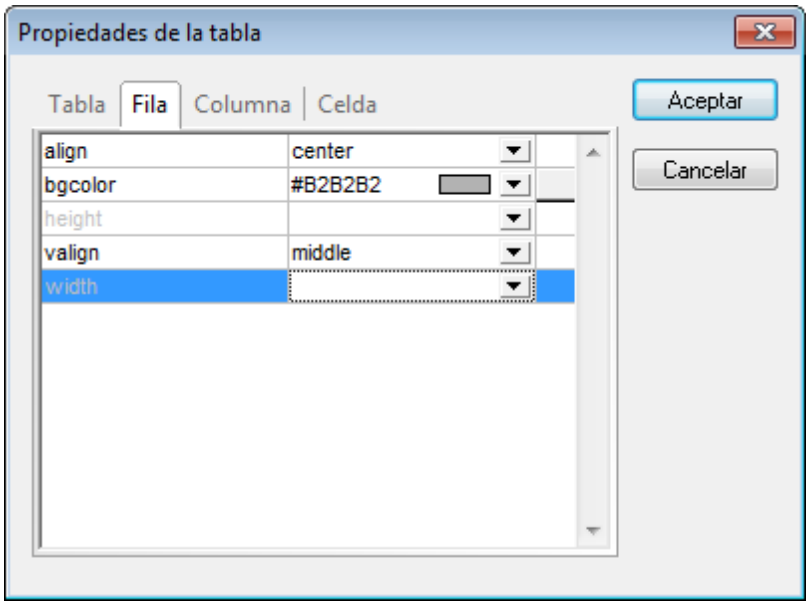
1. Ponga el cursor dentro de la tabla y haga clic en el icono (Propiedades de la tabla). Aparece el cuadro de diálogo "Propiedades de la tabla" (*imagen siguiente*), donde puede indicar el formato de la tabla, de una fila, de una columna o de una celda.



 © 2015 Altova GmbH

2. Defina un valor "0" para las propiedades `cellpadding` y `cellspacing` y haga clic en **Aceptar**. La tabla tendrá este aspecto:

3. Ponga el cursor en la primera fila y haga clic en el icono (Propiedades de la tabla). Aparece el cuadro de diálogo "Propiedades de la tabla". Haga clic en la pestaña **Fila**.



Vamos a usar la primera fila de la tabla como encabezado así que vamos a darle un color diferente para diferenciarla de las demás filas. Configure las propiedades de la primera fila tal y como muestra la imagen anterior y después escriba el texto de encabezado de cada columna. La tabla debería tener este aspecto:

Nombre	Teléfono	Correo electrónico

Observe que el texto está alineado en el centro, tal y como indica la propiedad `align`.

4. Ahora vamos a dividir la columna "Teléfono" en dos subcolumnas: "Oficina" y "Particular". Para ello debemos dividir el ancho horizontal de la columna "Teléfono" en dos columnas. Pero antes debemos dividir la celda de encabezado en dos filas. Ponga el cursor en la celda "Teléfono" y haga clic en el icono (Dividir verticalmente). La tabla debería tener este aspecto:

Nombre	Teléfono	Correo electrónico

5. Ahora ponga el cursor en la celda situada debajo de la celda "Teléfono" y haga clic en el

icono (Dividir horizontalmente). Escriba los encabezados de columna "Oficina" y "Particular". La tabla debería tener este aspecto:

Nombre	Teléfono		Correo electrónico
	Oficina	Particular	

El siguiente paso es dividir horizontalmente cada celda de la columna "Teléfono".

También puede agregar y eliminar columnas y filas, así como alinear verticalmente el contenido de las celdas, con los iconos de edición para tablas. Los iconos se describen en el apartado siguiente, titulado Iconos de edición para tablas CALS/HTML.

Desplazarse por las celdas de la tabla
Para desplazarse por las celdas de la tabla CALS/HTML utilice las flechas de dirección Arriba, Abajo, Derecha e Izquierda.

Introducir datos en una celda
Para introducir datos en una celda, ponga el cursor en la celda y teclee los datos.

Formato de texto
Al texto de las tablas CALS/HTML, al igual que a todo el texto del documento XML, se le da formato a través de elementos y atributos XML. Para añadir un elemento, resalte el texto y haga doble clic en el elemento pertinente en el ayudante de entrada Elementos. Para indicar un valor de atributo, ponga el cursor dentro del fragmento de texto y escriba el valor de atributo en el ayudante de entrada Atributos. Por ejemplo, puede poner en negrita el texto del encabezado de la tabla del ejemplo. Para ello resalte el texto y haga doble clic en el elemento `strong` en el ayudante de entrada Elementos. Para el elemento `strong` existe una plantilla global cuyo peso de fuente es `bold`.

Nota importante: para que el formato de texto aparezca en la vista Authentic, es necesario crear en StyleVision una plantilla global con formato de texto para el elemento en cuestión.

© 2015 Altova GmbH

4.2.3 Iconos de edición para tablas CALS/HTML

En este apartado describimos los comandos necesarios para editar tablas CALS/HTML, que están disponibles en forma de iconos en la barra de herramientas. Estos comandos no están disponibles en ningún menú.

Para más información sobre cómo usar estos iconos, consulte el apartado Tablas CALS/HTML.

Insertar tabla

Este comando inserta una **tabla CALS/HTML** en la posición del cursor.

Eliminar tabla

Este comando elimina la tabla activa.

Anexar fila

Este comando anexa una fila al final de la tabla activa.

Anexar columna

Este comando anexa una columna al final de la tabla activa.

Insertar fila

Este comando inserta una fila encima de la posición del cursor en la tabla activa.

Insertar columna

Este comando inserta una columna a la izquierda de la posición del cursor en la tabla activa.

Combinar celda hacia la izquierda

Este comando une la celda actual (donde se encuentra el cursor) con la celda de la izquierda. La celda nueva incluye las etiquetas de ambas celdas, el contenido de las celdas se encadena y los encabezados siguen como estaban.

Combinar celda hacia la derecha

Este comando une la celda actual (donde se encuentra el cursor) con la celda de la derecha. El contenido de las celdas se encadena en la celda nueva.

Combinar celda hacia abajo

Este comando une la celda actual (donde se encuentra el cursor) con la celda situada justo debajo. El contenido de las celdas se encadena en la celda nueva.

Combinar celda hacia arriba

Este comando une la celda actual (donde se encuentra el cursor) con la celda situada justo encima. El contenido de las celdas se encadena en la celda nueva.

Dividir horizontalmente

Este comando crea una celda nueva a la derecha de la celda activa. El tamaño de las celdas resultantes es el mismo que el de la celda original.

Dividir verticalmente

Este comando crea una celda nueva debajo de la celda activa.

Alinear arriba

Este comando alinea el contenido en la parte superior de la celda.

Centrar verticalmente

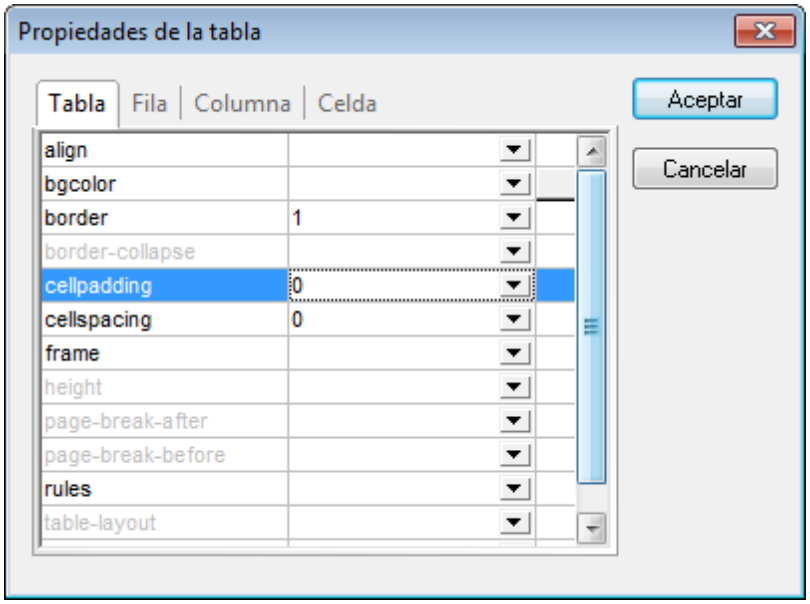
Este comando alinea el contenido en el centro de la celda.

Alinear abajo

Este comando alinea el contenido en la parte inferior de la celda.

Propiedades de la tabla

Este comando abre el cuadro de diálogo "Propiedades de la tabla". Este icono solo está habilitado para las tablas HTML, no para las tablas CALS.



© 2015 Altova GmbH

4.3 Editar una BD

En la vista Authentic puede editar tablas de base de datos (BD) y guardarlas en la BD. Este apartado ofrece una descripción de las funciones disponibles mientras se edita una tabla de BD. Antes de continuar, es necesario señalar estos aspectos generales:

- El diseñador de la hoja de estilos SPS puede limitar el número de registros de una tabla de BD que se presentan en la vista Authentic para que el diseño tenga un aspecto más compacto. Si así fuera, en la vista Authentic se carga únicamente ese número de registros. El usuario puede cargar y visualizar los demás registros de la tabla de BD con ayuda de los iconos de navegación (véase *Navegar por una tabla de BD*).
- Puede consultar la BD para ver determinados registros.
- Puede agregar, modificar y eliminar registros de la BD y guardar los cambios en la BD. Consulte el apartado Modificar una tabla de BD.

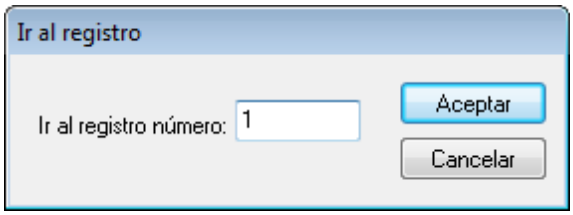
Note: En la vista Authentic, los datos que proceden de la base de datos de SQLite no son editables. cuando intente guardar los datos SQLite de la vista Authentic, un cuadro de mensaje le informará de esta limitación conocida.

.

4.3.1 Navegar por una tabla de BD

Los comandos necesarios para navegar por las filas de la tabla de BD están disponibles en forma de botones en el documento de la vista Authentic. Por lo general, cada tabla de BD viene acompañada de un panel de navegación que incluye cuatro o cinco botones.

Los iconos, de izquierda a derecha, significan: Ir al primer registro de la tabla de BD; Ir al registro anterior; Abrir el cuadro de diálogo "Ir al registro" (*ver imagen siguiente*); Ir al registro siguiente; e Ir al último registro de la tabla.



Pulse el botón pertinente para navegar por la tabla de la BD.

Bases de datos XML

En caso de trabajar con bases de datos XML, como IBM DB2, en las que una celda (o fila) contenga un documento XML, en la vista Authentic se cargan una por una las filas de la BD. Para cargar un documento XML que esté en otra fila, use el comando de menú **Authentic | Seleccionar una fila nueva con datos XML**.

© 2015 Altova GmbH

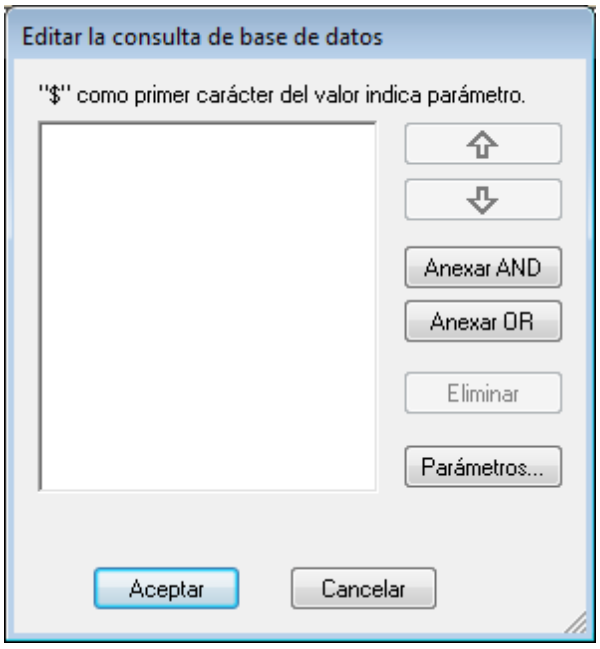
4.3.2 Consultas de BD

Las consultas de BD sirven para consultar los registros de una tabla de la vista Authentic. Por cada tabla se puede realizar una sola consulta y por cada consulta se puede consultar una sola tabla. Si al enviar la consulta hubiese cambios sin guardar en el documento de la vista Authentic, se le preguntará si desea guardar **todos** los cambios realizados en el documento o descartar **todos** los cambios. Tenga en cuenta que se guardarán/descartarán los cambios realizados en todas las tablas. Después de enviar la consulta la tabla se recarga utilizando las condiciones de la consulta.

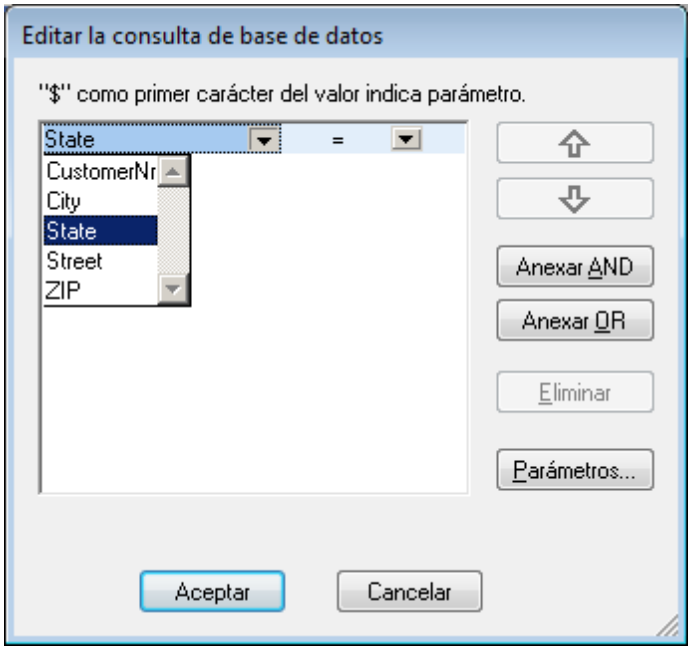
Nota importante: si recibe un mensaje diciendo que hay demasiadas tablas abiertas, puede reducir el número de tablas que están abiertas con ayuda de una consulta.

Para crear y enviar una consulta:

1. Haga clic en el botón de consulta para abrir el cuadro de diálogo "Editar la consulta de la base de datos" (*ver imagen*). Este botón suele aparecer encima de cada tabla de BD o justo debajo. Si no hay un botón de consulta para su tabla, significa que el diseñador de la hoja de estilos SPS no habilitó la función de consulta de BD para esa tabla.



2. Haga clic en el botón **Anexar AND** o **Anexar OR**. Estos botones anexan un criterio vacío para la consulta (*imagen siguiente*).



4. Escriba la expresión para el criterio de la consulta. Las expresiones están compuestas por (i) un nombre de campo (puede elegir uno en la lista del cuadro combinado), (ii) un operador (puede elegir uno en la lista del cuadro combinado) y (iii) un valor (que debe introducir a mano). Para más información sobre cómo construir expresiones, lea el subapartado Uso de expresiones en criterios de consulta.
5. Si desea añadir otro criterio, haga clic en los botones **Anexar AND** o **Anexar OR** según el operador lógico que desee utilizar para unir los dos criterios (AND o OR). Después añada el criterio nuevo. Para más información sobre los operadores lógicos, lea el subapartado Reordenar criterios en las consultas de BD.

Uso de expresiones en criterios de consulta

Las expresiones de los criterios de consulta de BD están formados por un nombre de campo, un operador y un valor. Los **nombres de campo disponibles** son los elementos secundarios de la tabla de datos de nivel superior seleccionada. Los nombres de estos campos aparecen en la lista del cuadro combinado (imagen anterior). Los **operadores** que puede usar en las expresiones son:

=	Igual a
<>	No igual
<	Menos que
<=	Menos o igual que
>	Más que
>=	Más o igual que
LIKE	Fonéticamente igual
NOT LIKE	Fonéticamente no igual
IS NULL	Está vacío
NOT NULL	No está vacío

Si selecciona la opción IS NULL o NOT NULL, el campo Valor se deshabilita. Los **valores** deben introducirse sin comillas (ni ningún otro delimitador). También es necesario que los valores tengan el mismo formato que su campo de BD correspondiente. De lo contrario, la expresión dará FALSE

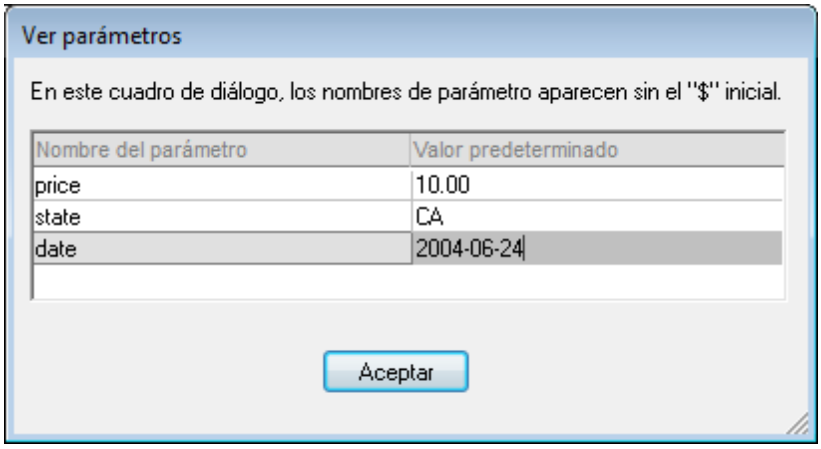
 © 2015 Altova GmbH

como resultado. Por ejemplo, si un criterio de consulta para un campo del tipo de datos `date` de una BD de MS Access tiene una expresión `StartDate=25/05/2004`, la expresión dará `FALSE` como resultado porque el tipo de datos `date` de las BD de MS Access tiene el formato `YYYY-MM-DD`.

Usar parámetros en consultas de BD

En su consulta puede introducir el nombre de un **parámetro** como valor de una expresión. Los parámetros son variables que se pueden usar en las consultas en lugar de valores literales. Cuando introduzca un parámetro en una expresión, en dicha expresión se usará el valor del parámetro. Los parámetros disponibles son los que definió el diseñador de la hoja de estilos en el archivo SPS. Los parámetros disponibles se enumeran en el cuadro de diálogo "Ver parámetros" (*imagen siguiente*). A los parámetros se les asigna un valor predeterminado en el archivo SPS. Este valor se puede reemplazar pasando otro valor al parámetro por medio de la línea de comandos (siempre y cuando el documento de salida se compile por medio de la línea de comandos).

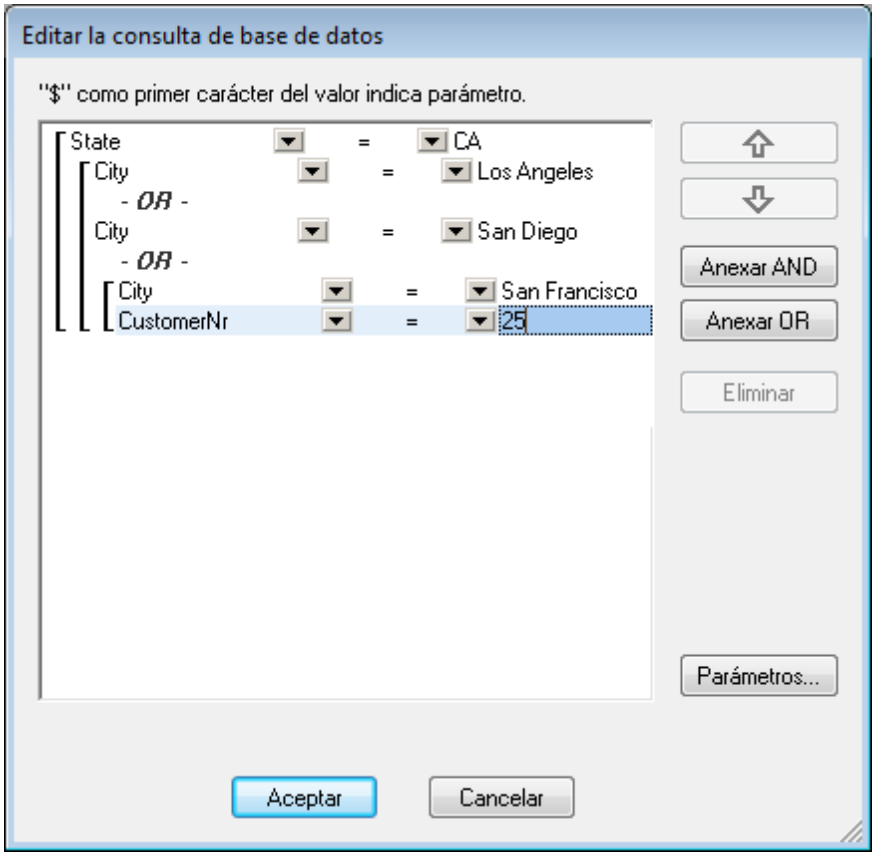
Para ver qué parámetros se definieron en el archivo SPS pulse el botón **Parámetros** del cuadro de diálogo "Editar la consulta de base de datos". Como resultado aparece el cuadro de diálogo "Ver parámetros" (*imagen siguiente*).



El cuadro de diálogo "Ver parámetros" incluye todos los parámetros que se definieron para la hoja de estilos en el archivo SPS. Los parámetros se pueden editar en el diseño de la hoja de estilos.

Reordenar criterios en las consultas de BD

La estructura lógica de la consulta de BD y la relación existente entre los criterios se indica de forma gráfica. Cada nivel de la estructura lógica se indica con un corchete. Los criterios o conjuntos de criterios adyacentes van unidos por el operador AND, mientras que si dos criterios están separados con la palabra `OR`, entonces se usa el operador OR. A los criterios también se aplica sangría para ofrecer una visión de conjunto de la estructura lógica de la consulta de la BD.



La consulta de BD de la imagen anterior se leería así:

```
State=CA AND (City=Los Angeles OR City=San Diego OR (City=San
Francisco AND CustomerNr=25))
```

Puede reordenar la consulta subiendo o bajando un criterio o conjunto de criterios. Para mover un criterio o conjunto de criterios, siga estos pasos:

1. Para seleccionar un criterio, haga clic en él. Para seleccionar un nivel entero, haga clic en el corchete que abarca dicho nivel.
2. Haga clic en uno de los dos botones en forma de flecha según corresponda.

Antes de subir o bajar un criterio debe tener en cuenta varias cuestiones:

- Si el criterio situado justo encima o debajo está al mismo nivel, el criterio elegido toma la posición de otro y viceversa.
- Si sube o baja un conjunto de criterios (es decir, criterios que están dentro de un corchete), el conjunto cambia de posición dentro del mismo nivel, no cambia de nivel.
- Si el criterio situado justo encima o debajo está en otro nivel (superior o inferior), el criterio seleccionado pasará al siguiente nivel (superior o inferior), **nivel por nivel**.

Para eliminar un criterio de una consulta de BD, selecciónelo y haga clic en **Eliminar**.

Modificar una consulta de BD

Para modificar una consulta de BD:

1. Haga clic en el botón de consulta . Se abre el cuadro de diálogo "Editar la consulta

 © 2015 Altova GmbH

de base de datos". Ahora puede editar las expresiones de los criterios, agregar criterios nuevos, reordenarlos o eliminarlos.

2. Haga clic en **Aceptar**. Lo datos de la BD se recargan automáticamente en la vista Authentic, para reflejar los cambios realizados en la consulta de BD.

4.3.3 Modificar una tabla de BD

Agregar un registro
Para agregar un registro a una tabla de BD:

1. Ponga el cursor en la fila de la tabla de BD y haga clic en el icono (para anexar una fila) o en (para insertar una fila). Como resultado se crea un registro nuevo en el archivo XML temporal.
2. Haga clic en el comando **Archivo | Guardar** para agregar el registro nuevo en la BD. En la vista Authentic se anexa una fila nueva para el registro nuevo en la tabla de BD. El valor del campo `AltovaRowStatus` de este registro pasa a ser `A` (de "Added", es decir, agregado).

Los datos que introduzca para el registro nuevo aparecen en negrita y subrayados. Así puede distinguir los registros agregados de los registros ya existentes (siempre que no se aplicase ese mismo formato a ningún registro ya existente). Los errores de tipos de datos aparecen en color rojo para que pueda detectarlos inmediatamente.

El nuevo registro se agrega a la BD nada más hacer clic en **Archivo | Guardar**. Una vez agregado el nuevo registro a la BD, su campo `AltovaRowStatus` se inicializa (se indica con `---`) y el registro aparece en la vista Authentic como cualquier otro registro.

Modificar un registro
Para modificar un registro, ponga el cursor en el registro de la tabla de BD y edítelo. Si se limitó el número de registros que se pueden mostrar, entonces quizás tenga que navegar hasta el registro (véase Navegar por una tabla de BD).

Cuando se modifica un registro, las entradas de todos los campos del registro se subrayan y el valor del campo `AltovaRowStatus` de las instancias principales de este registro pasa a ser `U` (de "Updated", es decir, actualizado). En las instancias secundarias de este registro el valor del campo `AltovaRowStatus` pasa a ser `u` (en minúsculas). Las instancias principales y secundarias de un registro están definidas por la estructura de la BD (y del esquema XML generado a partir de ella). Por ejemplo, si se incluye una tabla Dirección en una tabla Cliente, entonces la tabla Dirección puede aparecer en el documento de diseño en dos tipos de instancias: como tabla Dirección propiamente dicha y dentro de las instancias de la tabla Cliente. El tipo de instancia que se modifique es el tipo principal. Los demás tipos son los tipos secundarios. Los errores de tipo de datos aparecen en color rojo para que pueda detectarlos inmediatamente.

Los cambios se guardan en la BD al hacer clic en **Archivo | Guardar**. Después de guardar en la BD un registro en el que se realizaron cambios, su campo `AltovaRowStatus` se inicializa (se indica con `---`) y el registro aparece en la vista Authentic como cualquier otro registro.

Nota importante:

- Si se modifica un solo campo del registro en la vista Authentic, al guardar los datos en la BD se actualiza el registro entero.
- El valor de fecha `0001-01-01` se define como valor `NULL` en algunas BD, lo cual podría dar lugar a errores.

Eliminar un registro
Para eliminar un registro:

 © 2015 Altova GmbH

1. Ponga el cursor en la fila del registro que desea eliminar y haga clic en el icono . El registro que se va a eliminar aparece tachado. El campo `AltovaRowStatus` toma el valor `D` en las instancias principales del registro, `d` en las instancias secundarias y `X` en los registros eliminados de forma indirecta. Los registros eliminados de forma indirecta son aquellos campos del registro eliminado que se almacenan en otra tabla diferente. Por ejemplo, la tabla Dirección puede estar incluida en una tabla Cliente. Si se elimina un registro de Cliente, su registro Dirección correspondiente se elimina de forma indirecta. Si se elimina un registro Dirección de la tabla Cliente, entonces el registro Dirección de la tabla Cliente se elimina de forma principal, pero el mismo registro se elimina de forma secundaria en la tabla independiente Dirección si se creó una instancia de esta tabla.
2. Haga clic en **Archivo | Guardar** para guardar los cambios en la BD.

Nota importante: después de guardar los cambios en la BD el comando **Deshacer** se restablece. Es decir, ya no podrá deshacer las acciones llevadas a cabo antes de guardar los cambios.

4.4 Trabajar con fechas

En la vista Authentic las fechas se pueden editar de dos maneras diferentes:

- Con ayuda del selector de fecha.
- O escribiendo el valor de la fecha a mano.

El método al que tiene acceso el usuario de la vista Authentic lo define el diseñador de la hoja de estilos en el archivo SPS. En los siguientes apartados describimos ambos métodos más detalladamente.

Nota sobre los formatos de fecha

En el documento XML las fechas se almacenan en un tipo de datos de fecha. Para que el documento XML sea válido, la fecha debe almacenarse en un formato léxico concreto, dependiendo del tipo de datos. Por ejemplo, para el tipo de datos `xs:date` la fecha debe tener el formato léxico `YYYY-MM-DD`. Si en un nodo con el tipo de datos `xs:date` introduce una fecha que no se ajusta a este formato, el documento XML no será válido.

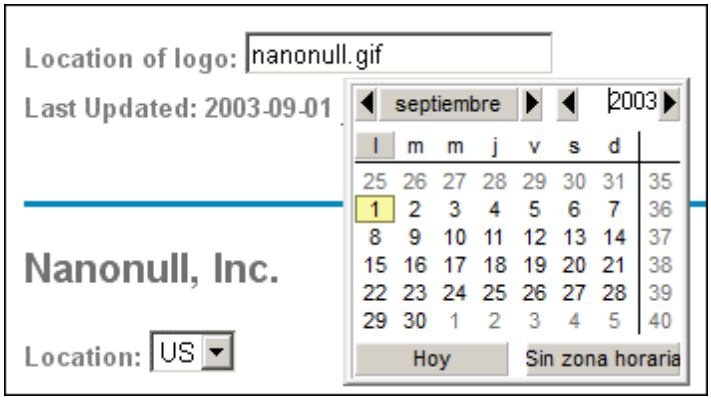
Para garantizar que las fechas se introduzcan en el formato correcto, el diseñador del SPS puede incluir en el diseño un selector de fecha gráfico. Así se asegura de que la fecha seleccionada en el selector de fecha se introduce en el formato léxico correcto. Si no hay un selector de fecha, el usuario de la vista Authentic debe validar el documento XML para comprobar si todas las fechas introducidas tienen el formato léxico correcto.

© 2015 Altova GmbH

4.4.1 Selector de fecha

El selector de fecha es un calendario gráfico que sirve para introducir fechas en un formato estándar en el documento XML. Es importante tener un formato estándar para procesar los datos del documento. El icono del selector de fecha aparece junto al campo de fecha al que modifica (*ver imagen*).

Para abrir el selector, haga clic en el icono del selector de fecha (*ver imagen*).

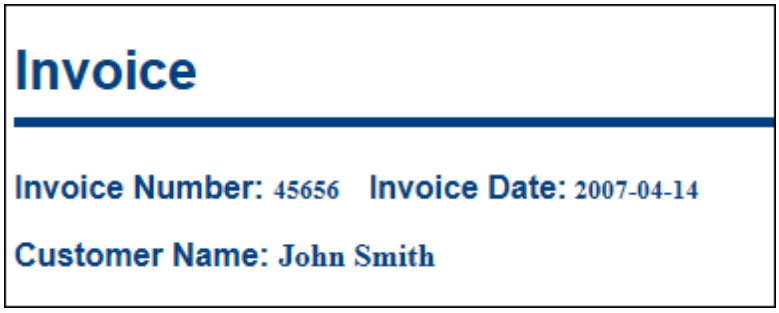


Para seleccionar una fecha, haga clic en la fecha, el mes y el año. Los datos introducidos en el documento XML y los datos de la vista Authentic se actualizan en función de la fecha seleccionada en el selector. Si lo desea también puede introducir la zona horaria.

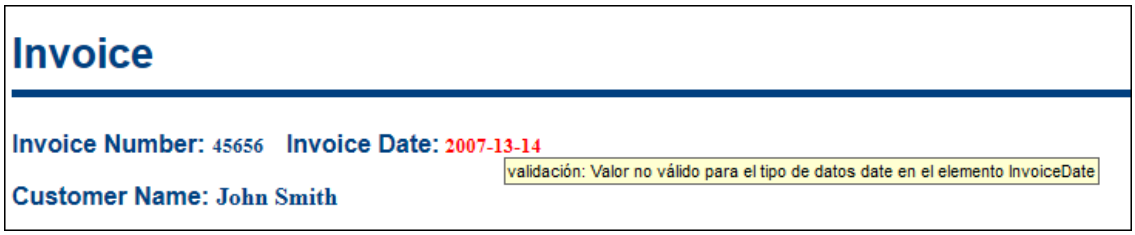
4.4.2 Entrada de texto

En los campos de fecha que no tengan un selector de fecha (*ver imagen*), puede editar la fecha directamente con solo teclear la fecha nueva.

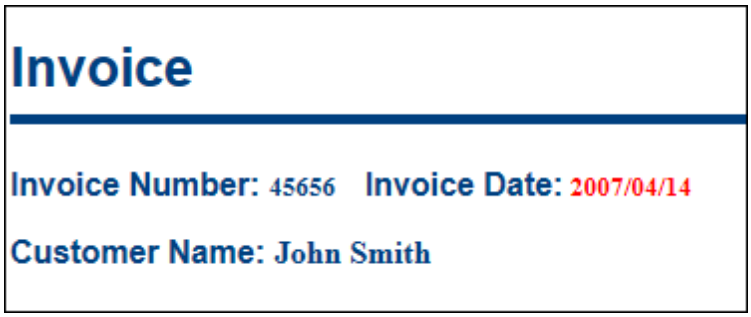
Nota: no cambie el formato al cambiar la fecha.



Si edita una fecha y la fecha nueva no está dentro del rango válido, la fecha se vuelve de color rojo para informarle del error. Si pasa el puntero del mouse sobre la fecha no válida, aparece un mensaje de error (*ver imagen*).


Invoice
Invoice Number: 45656 Invoice Date: 2007-13-14
validación: Valor no válido para el tipo de datos date en el elemento InvoiceDate
Customer Name: John Smith

Si intenta cambiar el formato de la fecha, la fecha se vuelve de color rojo para informarle del error (*ver imagen*).



© 2015 Altova GmbH

4.5 Definir entidades

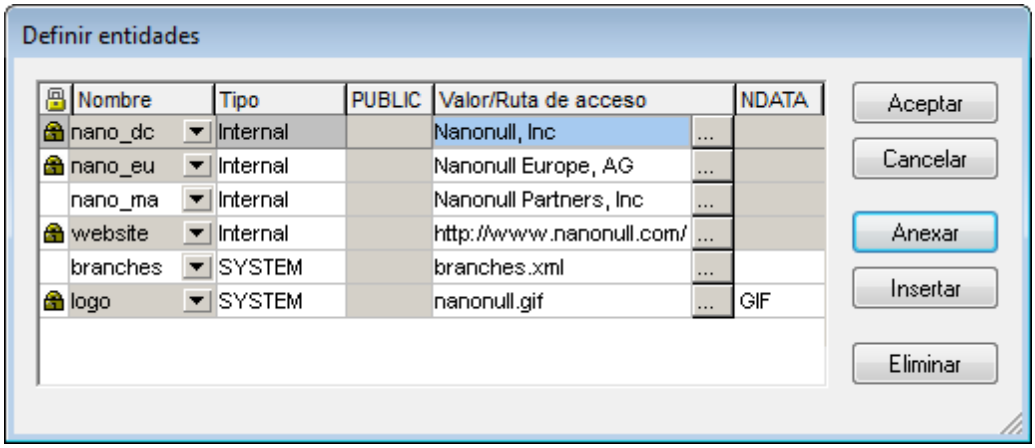
Puede definir las entidades que se deben usar en la vista Authentic, tanto si el documento está basado en una DTD como en un esquema XML. Una vez definidas, las entidades aparecen en el ayudante de entrada Entidades y en el submenú del menú contextual. Al hacer doble clic en una entidad del ayudante de entrada Entidades, la entidad se inserta en el punto de inserción del cursor.

Puede usar entidades para las cadenas de texto, fragmentos XML y recursos externos que vaya a utilizar repetidas veces en el documento. Las entidades (que no son más que una abreviatura de los datos) se definen en el cuadro de diálogo "Definir entidades". Una vez definidas, puede usarlas tantas veces como quiera en el documento, lo cual permite ahorrar tiempo y mantener mejor el documento.

Hay dos tipos principales de entidades: las **entidades analizadas**, que son datos XML (ya sea una cadena de texto o un fragmento de un documento XML) y las **entidades sin analizar**, que son datos no XML, como por ejemplo archivos binarios (normalmente objetos gráficos, sonoros o multimedia). Cada entidad tiene un nombre y un valor. En el caso de las entidades analizadas, la entidad es un marcador de posición para los datos XML. El valor de la entidad son los datos XML propiamente dichos o un URI que apunta a un archivo XML que contiene los datos XML. En el caso de las entidades sin analizar, el valor de la entidad es un URI que apunta al archivo no XML.

Para definir una entidad:

1. Haga clic en **Authentic | Definir entidades XML...**. Se abre el cuadro de diálogo "Definir entidades" (*ver imagen*).



2. Escriba el nombre de la entidad en el campo Nombre. Es el nombre que aparecerá en el ayudante de entrada Entidades.
3. Seleccione el tipo de entidad en la lista desplegable del campo Tipo. Los tipos son: **Internal** para entidades cuyo texto está almacenado en el documento XML propiamente dicho. **PUBLIC** y **SYSTEM** indican que el recurso se encuentra fuera del archivo XML y que se localizará por medio de un identificador público o de sistema respectivamente. Un identificador de sistema es un URI que da la ubicación del recurso. Un identificador público es un identificador independiente de la ubicación, que permite a los procesadores identificar el recurso. Si indica tanto un identificador público como de sistema, el identificador público se convierte en identificador de sistema y se usa el identificador de sistema.
4. Si selecciona el tipo PUBLIC, escriba el identificador público del recurso en el campo PUBLIC. Si selecciona el tipo Internal o SYSTEM, entonces se deshabilita el campo

PUBLIC.

5. En el campo Valor/Ruta de acceso puede introducir una de estas tres opciones:
   - Si el tipo de entidad es Internal, escriba la cadena de texto que desea usar como valor de la entidad. No escriba comillas para delimitar el texto. Si introduce comillas, estas se tratarán como si fueran parte de la cadena de texto.
   - Si el tipo de entidad es SYSTEM, escriba el URI del recurso o seleccione un recurso de su red local (con el botón Examinar). Si el recurso contiene datos analizados, entonces debe ser un archivo XML (con la extensión de archivo `.xml`). También puede ser un archivo binario, como un archivo GIF.
   - Si el tipo de entidad es PUBLIC, entonces debe introducir un identificador de sistema en el campo PUBLIC.
6. Los datos introducidos en el campo NDATA indican al procesador que esta entidad no se debe analizar, sino que debe enviarse al procesador adecuado. El campo NDATA debe contener un valor que indique que la entidad es una entidad sin analizar.

Características del cuadro de diálogo

En el cuadro de diálogo "Definir entidades" el usuario puede:

- Anexar entidades.
- Insertar entidades.
- Eliminar entidades.
- Ordenar entidades por orden alfabético en base al contenido de cualquiera de las columnas. Para ello haga clic en el encabezado de la columna pertinente. Un clic ordena el contenido de la columna en orden ascendente, dos clics ordenan el contenido en orden descendente.
- Cambiar el tamaño del cuadro de diálogo y el ancho de las columnas.
- Bloqueo. Cuando se usa una entidad en el documento XML, esta se bloquea y ya no se puede editar en el cuadro de diálogo "Definir entidades". Las entidades bloqueadas se identifican con un símbolo en forma de candado que aparece en la primera columna del cuadro de diálogo. Este bloqueo garantiza la validez del documento XML, en lo que a las entidades se refiere. (El documento no sería válido si se hace referencia a una entidad pero esta no está definida.)
- Las entidades repetidas aparecen marcadas.

Limitaciones de las entidades

- Las entidades que están dentro de otra entidad no se resuelven, ya sea en el diálogo, en la vista Authentic o en el resultado XSLT, y el carácter et de dicha entidad aparece como caracteres de escape, es decir `&`.
- Las entidades sin analizar externas que no sean archivos de imagen no se resuelven en la vista Authentic. Si se define una imagen del diseño para que lea una entidad externa sin analizar y el URI de la imagen se define como un nombre de entidad ("`logo`", por ejemplo), entonces este nombre de entidad se puede definir en el cuadro de diálogo "Definir entidades" (*imagen anterior*) como entidad externa sin analizar con un valor que se convierte en el URI del archivo de imagen (como en el caso de la entidad `logo` de la imagen).

© 2015 Altova GmbH

4.6 Imágenes en la vista Authentic

En la vista Authentic puede indicar qué imágenes se deben usar en el documento final de salida (en HTML, RTF, PDF o Word 2007). Tenga en cuenta que es posible que algunos formatos de imagen no sean compatibles con todos los formatos de salida ni con todas las aplicaciones. Por ejemplo, el formato SVG es compatible con PDF, pero no con RTF. Además, para ver este formato en HTML es necesario instalar un complemento para el explorador. Por tanto, cuando seleccione el formato de la imagen, seleccione un formato que sea compatible con el formato de salida de su documento. La mayoría de formatos de imagen son compatibles con todos los formatos de salida (*consulte la lista que aparece más abajo*).

La vista Authentic se basa en Internet Explorer y puede mostrar la mayoría de los formatos de imagen que Internet Explorer puede mostrar. Estos son los formatos de imagen compatibles con la vista Authentic:

- GIF
- JPG
- PNG
- BMP
- WMF (Microsoft Windows Metafile)
- EMF (Enhanced Metafile)
- SVG (sólo para documentos de salida en PDF)

Rutas de acceso relativas

Las rutas de acceso relativas se resuelven en relación al archivo SPS.

4.7 Teclas de acceso rápido en la vista Authentic

Tecla Entrar
En la vista Authentic la tecla **Entrar** se usa para anexar elementos adicionales cuando el cursor está situado en ciertas posiciones. Por ejemplo, si el capítulo de un libro puede tener varios párrafos (porque el esquema lo permite), al pulsar la tecla **Entrar** dentro del texto de un párrafo se crea un párrafo nuevo inmediatamente después del párrafo actual. Si el capítulo puede tener un título y varios párrafos, al pulsar la tecla **Entrar** dentro del capítulo pero no dentro de un párrafo (incluido el elemento Título), se crea un capítulo nuevo justo después del capítulo actual (siempre y cuando el esquema autorice varios capítulos).

Nota: la tecla **Entrar no inserta** una línea nueva. Incluso cuando el cursor está dentro de un nodo de texto, como un párrafo por ejemplo.

Uso del teclado

Puede usar el teclado igual que siempre, tanto para escribir como para navegar. No obstante, debe tener en cuenta que:

- La tecla **Tabulador** mueve el cursor hacia adelante, deteniéndose antes y después de los nodos y resaltando su contenido. Sin embargo, el cursor no se detiene en contenido estático.
- Los hipervínculos `agregar...` y `agregar Nodo` se consideran nodos de contenido y se resaltan al llegar a ellos con la tecla **Tabulador**. Para activar estos nodos, pulse la barra espaciadora o la tecla **Entrar**.

© 2015 Altova GmbH

5 Vista Explorador

La vista **Explorador** se suele usar para ver:

- Archivos XML que tienen un archivo XSLT asociado. Al cambiar a la vista **Explorador**, el archivo XML se transforma instantáneamente usando la hoja de estilos XSLT asociada y el resultado se presenta directamente en la vista **Explorador**.
- Archivos HTML que se crearon directamente como HTML o a partir de un archivo XML transformado con una hoja de estilos XSLT.

Para ver archivos XML y HTML en la vista **Explorador**, basta con hacer clic en la pestaña **Explorador**.

Nota sobre Microsoft Internet Explorer y XSLT

Para usar la vista **Explorador** es necesario tener instalado Microsoft Internet Explorer 5.0 o superior. Si desea usar la vista Explorador para ver archivos XML transformados con una hoja de estilos XSLT, recomendamos instalar Internet Explorer 6.0 o superior. Estas versiones de Internet Explorer utilizan MSXML 3.0, un analizador XML totalmente compatible con el estándar XSLT 1.0. Si lo desea, también puede instalar MSXML 4.0. Para más información, visite nuestro centro de descargas. (Recuerde que IE 5 no es 100% compatible con la recomendación XSLT oficial. De modo que si tiene problemas en la vista Explorar con IE 5, rogamos instale IE 6 o superior.) También recomendamos consultar la compatibilidad de su versión de Internet Explorer con XSLT.

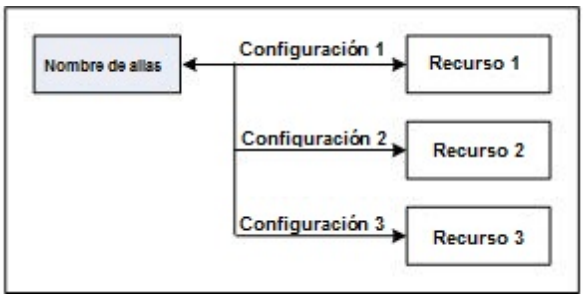
Funciones de la vista Explorador

Estas son las funciones disponibles en la vista **Explorador**. Para tener acceso a estas funciones, utilice los menús **Explorador**, **Archivo** y **Edición**.

- Abrir en una ventana separada: cuando la vista **Explorador** se abre en una ventana separada, se puede colocar en paralelo a una de las vistas de edición del mismo documento. Este comando está en el menú **Explorador** y se puede activar o desactivar (es decir, si hace clic otra vez en el comando, la ventana separada de la vista Explorador se restaura y vuelve a formar parte de la ventana principal). En la pestaña Vista del cuadro de diálogo "Opciones", puede especificar si la vista **Explorador** se debe abrir por defecto en una ventana separada o no.
- **Adelante** y **Atrás**: los comandos típicos de un explorador para navegar por las páginas que se cargaron en la vista **Explorador**. Estos comandos están en el menú **Explorador**.
- Tamaño de la fuente: se puede ajustar con el comando **Tamaño de fuente** del menú **Explorador**.
- **Detener**, **Actualizar** e **Imprimir**: más comandos típicos de un explorador. Estos están en los menús **Explorador** y **Archivo**.
- **Buscar**: permite buscar cadenas de texto. Este comando está en el menú **Edición**.

6 Recursos globales de Altova

Los recursos globales de Altova son un conjunto de alias para recursos de archivo, carpeta y base de datos. Cada alias puede tener varias configuraciones y cada configuración se corresponde con un solo recurso (*imagen siguiente*). Por tanto, cuando un recurso global se utiliza como entrada de datos, se puede alternar la configuración del recurso global utilizado. La interfaz gráfica de usuario ofrece sencillos controles para cambiar de configuración activa.



El uso de los recursos globales de Altova conlleva dos procesos:

- Definir los recursos globales: los recursos se definen y sus definiciones se almacenan en un archivo XML. Estos recursos se pueden compartir con varias aplicaciones de Altova.
- Usar los recursos globales: dentro de una aplicación de Altova puede localizar archivos con un recurso global en lugar de con una ruta de acceso. La ventaja es que puede cambiar de recurso con solo cambiar de configuración activa en Authentic Desktop.

Recursos globales en otros productos de Altova

Por el momento, puede definir y usar recursos globales en estos productos de Altova: XMLSpy, StyleVision, MapForce, Authentic Desktop, MobileTogether Designer y DatabaseSpy.

© 2015 Altova GmbH

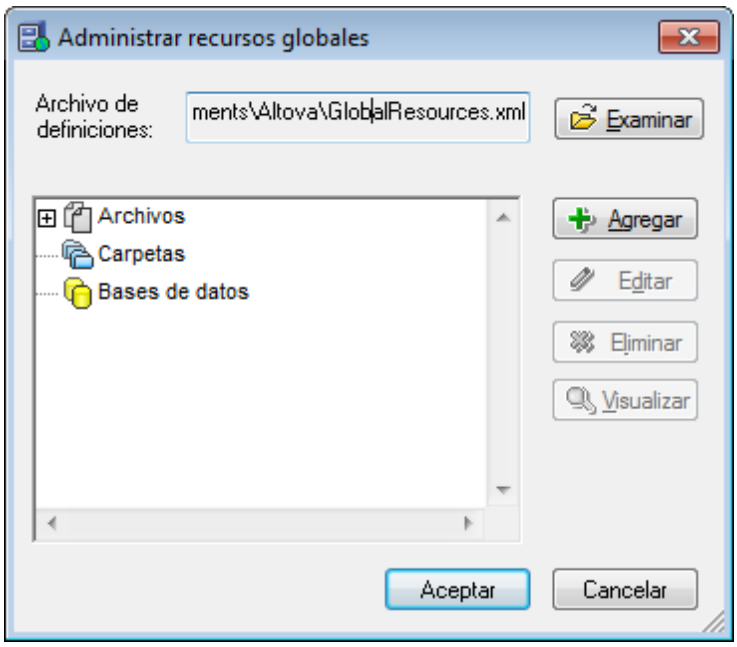
6.1 Definir recursos globales

Los recursos globales de Altova se definen en el cuadro de diálogo "Administrar recursos globales". Hay dos maneras de abrir este cuadro de diálogo:

- Seleccionando el comando de menú **Herramientas | Recursos globales**.
- Haciendo clic en el icono **Administrar recursos globales** de la barra de herramientas Recursos globales (*imagen siguiente*).

El archivo de definiciones de recursos globales

La información sobre los recursos globales se almacena en un archivo XML llamado archivo de definiciones de recursos globales. Este archivo se crea y guarda cuando se define el primer recurso global en el cuadro de diálogo "Administrar recursos globales" (*imagen siguiente*).



Cuando se abre el cuadro de diálogo "Administrar recursos globales" por primera vez, el nombre y la ubicación predeterminados del archivo de definiciones de recursos globales aparece en el cuadro de texto *Archivo de definiciones* (*imagen anterior*):

```
C:\Users\<nombreUsuario>\Mis Documentos\Altova\GlobalResources.xml
```

Este archivo se define como archivo predeterminado de definiciones de recursos globales para todas las aplicaciones de Altova. Así puede guardar en este archivo recursos globales desde cualquier aplicación de Altova y el recurso global estará a disposición de todas las aplicaciones de Altova. Para definir y guardar un recurso global en el archivo de definiciones, añada el recurso global en el cuadro de diálogo "Administrar recursos globales" y haga clic en **Aceptar**.

Para seleccionar un archivo de definiciones de recursos globales ya existente como archivo de definiciones activo, búsquelo con el botón **Examinar** del cuadro de texto *Archivo de definiciones* (*imagen anterior*).

Nota: puede dar el nombre que quiera al archivo de definiciones y guardarlo en cualquier ubicación. Después en las aplicaciones de Altova sólo tiene que definir este archivo como archivo de definiciones de recursos globales (en el cuadro de texto *Archivo de definiciones*). Si usa un solo archivo de definiciones para todas las aplicaciones de Altova, los recursos definidos en el archivo estarán a disposición de todas las aplicaciones de Altova.

Nota: si lo prefiere, también puede crear varios archivos de definiciones de recursos globales. Sin embargo, en cada aplicación de Altova sólo puede estar activo un archivo de definiciones y, por tanto, sólo las definiciones de ese archivo estarán a disposición de la aplicación.

Administración de recursos globales: cómo añadir, editar, eliminar y guardar recursos

En el cuadro de diálogo "Administrar recursos globales" (*imagen anterior*) puede añadir un recurso global al archivo de definiciones seleccionado o editar/eliminar un recurso global. El archivo de definiciones de recursos globales organiza los recursos en tres grupos: archivos, carpetas y bases de datos (*ver imagen anterior*).

Para **añadir un recurso global**, haga clic en el botón **Agregar** y defina el recurso global en el cuadro de diálogo que aparece (*consulte los subapartados archivos, carpetas y bases de datos para más información*). Después de definir el recurso global y guardarlo (haciendo clic en **Aceptar** en el cuadro de diálogo "Administrar recursos globales"), el recurso global se añade a la biblioteca de definiciones globales del archivo de definiciones seleccionado. El recurso global se identificará por medio de un alias.

Para **editar un recurso global**, selecciónelo y haga clic en **Editar**. En el cuadro de diálogo que aparece puede realizar todos los cambios necesarios (*consulte los subapartados archivos, carpetas y bases de datos para más información*).

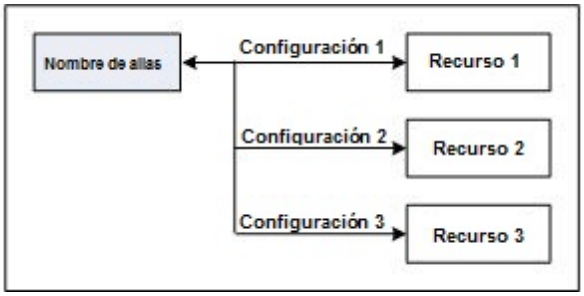
Para **eliminar un recurso global**, selecciónelo y haga clic en **Eliminar**.

Cuando termine de añadir, editar y eliminar recursos globales, es necesario pulsar el botón **Aceptar** del cuadro de diálogo "Administrar recursos globales" para que **los cambios se guarden** en el archivo de definiciones de recursos globales.

Relacionar recursos globales con nombres de alias mediante configuraciones

Definir un recurso global implica asignar un nombre de alias a un recurso (a un archivo, una carpeta o una base de datos). Además se puede asignar el mismo nombre de alias a varios recursos. Cada asignación de nombre de alias se denomina configuración. Por tanto, el mismo nombre de alias se puede asociar a varios recursos mediante diferentes configuraciones (*ver imagen siguiente*).

© 2015 Altova GmbH



Después en la aplicación de Altova puede asignar alias en lugar de archivos y cambiar de recursos (los recursos asignados a cada alias) con solo cambiar de configuración de recursos globales (configuración activa). Por ejemplo, imaginemos que en Altova XMLSpy desea ejecutar una transformación XSLT en el documento XML `MyXML.xml`. Pues bien, para ello puede asignar a este archivo el nombre de alias `MyXSLT` para utilizarlo como recurso global en las transformaciones XSLT. Después en XMLSpy puede cambiar de configuración activa para hacer uso de diferentes archivos XSLT. Si la configuración `Configuración-1` asigna el archivo `First.xslt` al nombre de alias `MyXSLT` y se selecciona `Configuración-1` como configuración activa, entonces se usará el archivo `First.xslt` para la transformación. De este modo, puede usar varias configuraciones para acceder a varios recursos que tengan el mismo alias. Este mecanismo puede ser muy práctico a la hora de probar y comparar recursos. Por otra parte, ya que los recursos globales se pueden usar en todos los productos de Altova, podrá probar y comparar sus recursos en varias herramientas.

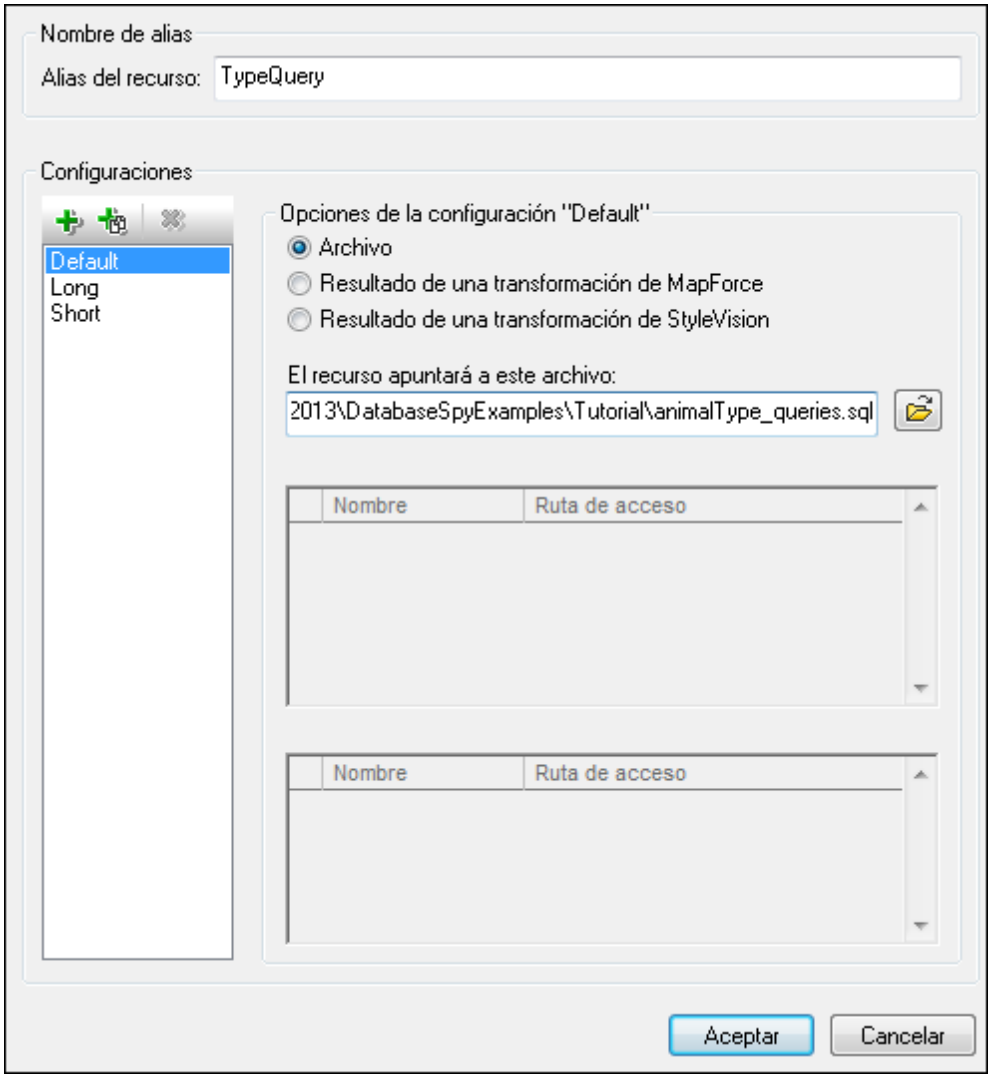
6.1.1 Archivos

El cuadro de diálogo "Recurso global" para archivos (*imagen siguiente*) se abre con el comando **Agregar | Archivos** del cuadro de diálogo Administrar recursos globales. En este cuadro de diálogo puede indicar las configuraciones del alias que aparece en el cuadro de texto *Alias del recurso*. Después de definir las propiedades de las configuraciones (tal y como se explica más abajo), guarde la definición del alias haciendo clic en **Aceptar**.

Tras guardar una definición de alias puede agregar otro alias repitiendo los pasos que acabamos de describir (empezando por el comando **Agregar | Archivos** del cuadro de diálogo Administrar recursos globales).

Cuadro de diálogo "Recursos globales"

Los alias se definen en el cuadro de diálogo de recursos globales (*imagen siguiente*).



© 2015 Altova GmbH

Iconos del cuadro de diálogo de recursos globales

Agregar configuración: abre el cuadro de diálogo "Agregar configuración", donde puede introducir el nombre de la configuración que se debe agregar.

Agregar una copia de configuración: abre el cuadro de diálogo "Agregar configuración", donde puede escribir el nombre de la configuración que debe crearse como copia de la configuración seleccionada.

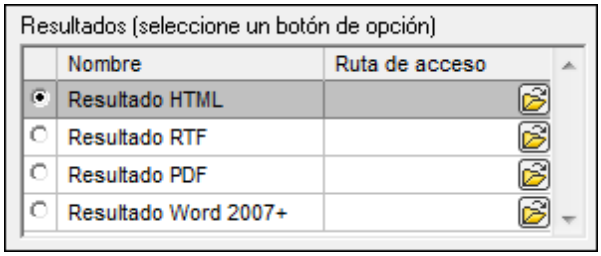
Eliminar: elimina la configuración seleccionada.

Abrir: examina el sistema de archivos para buscar el archivo que se debe usar para crear el recurso global.

Definir el alias

Siga estos pasos para definir el alias (el nombre de alias y sus configuraciones):

1. *Póngale un nombre al alias:* escriba el nombre del alias en el cuadro de texto *Alias del recurso.*
2. *Añada configuraciones:* el panel *Configuraciones* tiene por defecto una configuración llamada `Default` (*ver imagen anterior*), que no se puede editar ni eliminar. Puede agregar más configuraciones de este modo: (i) haga clic en los iconos **Agregar configuración** o **Agregar copia de configuración** y (ii) póngale un nombre a la configuración en el cuadro de diálogo que aparece. Cada configuración que añada aparecerá en la lista *Configuraciones*. En la imagen anterior, por ejemplo, se añadieron las configuraciones `Long` y `Short`. El comando **Agregar copia de configuración** permite copiar la configuración seleccionada y después modificarla.
3. *Seleccione un tipo de recurso para cada configuración:* seleccione una configuración de la lista *Configuraciones* y, en el panel *Opciones de la configuración*, especifique el tipo de recurso para esa configuración: puede escoger entre (i) un archivo, (ii) el resultado de una transformación de Altova MapForce y (iii) el resultado de una transformación de Altova StyleVision. Si selecciona una transformación de MapForce o de StyleVision, entonces la transformación se lleva a cabo con MapForce o StyleVision usando el archivo `.mfd`/`.sps` y el archivo de entrada correspondiente. El resultado de la transformación se usa como recurso global de la configuración.
4. *Seleccione un archivo para el tipo de recurso:* pulse el icono Examinar del cuadro *El recurso apuntará a....* y seleccione el archivo, el archivo .mfd (para transformaciones de MapForce) o el archivo .sps (para transformaciones de StyleVision). Si la transformación permite usar varias entradas y varios resultados, podrá elegir cuál de ellos se utiliza. Por ejemplo, dependiendo de qué edición de StyleVision tenga instalada, podrá elegir entre unos resultados u otros (en la imagen siguiente se ofrecen documentos de resultados de la edición Enterprise Edition).



Seleccione una opción (por ejemplo, en la imagen anterior se seleccionó el resultado en formato HTML). Si el recurso es el resultado de una transformación, el resultado se puede guardar como archivo o como recurso global. Haga clic en el icono y seleccione **Examinar...** (si desea guardar el resultado como archivo) o **Elegir otro recurso global...** (si desea guardar el resultado como recurso global). Si no selecciona ninguna de estas dos opciones, el resultado de la transformación se cargará como archivo temporal cuando se invoque el recurso global.

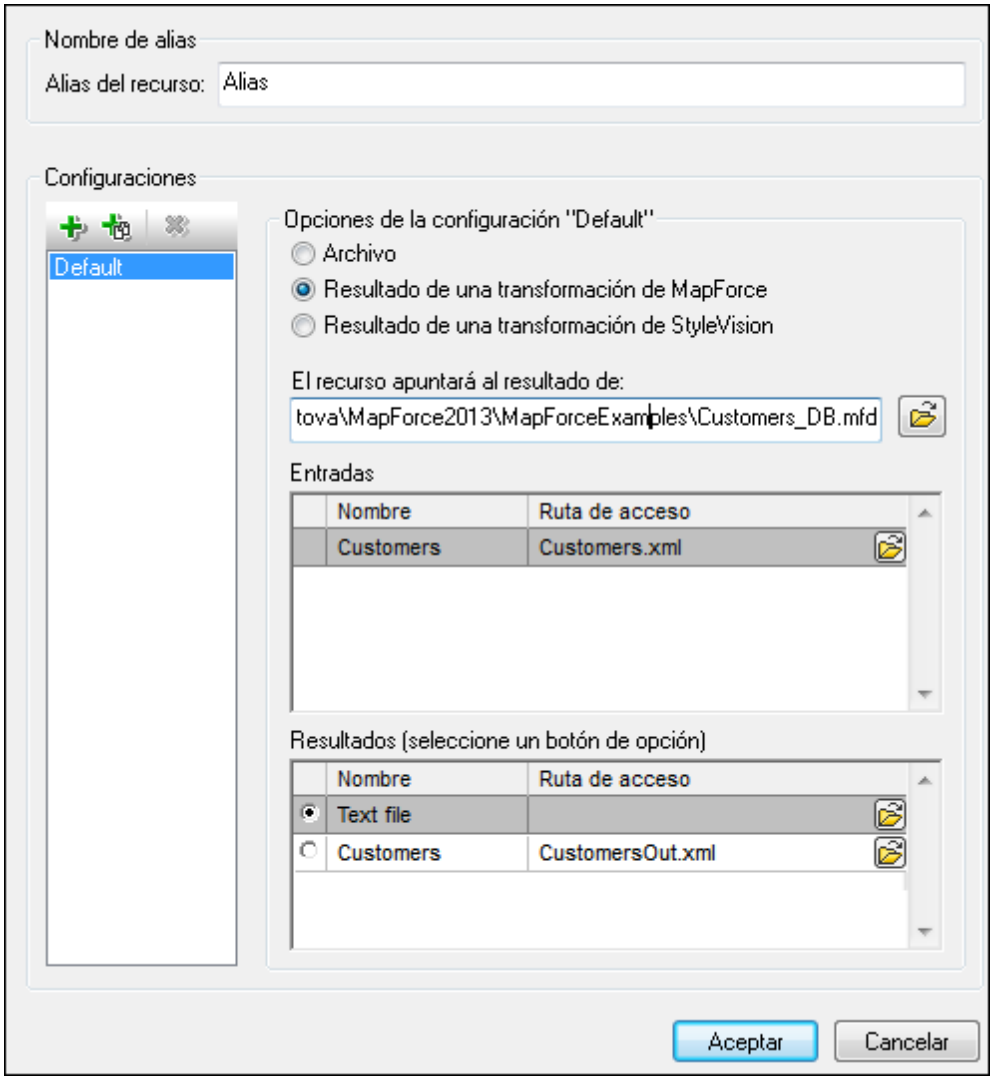
5. *Si es necesario, defina más configuraciones*: si quiere, puede añadir más configuraciones y especificar un recurso para cada una de ellas. Para ello repita los pasos 3 y 4. Puede agregar una configuración nueva a la definición de alias en todo momento.
6. *Guarde la definición de alias:* haga clic en **Aceptar** para guardar el alias y todas sus configuraciones como recurso global. El recurso global nuevo aparecerá bajo el nodo *Archivos* del cuadro de diálogo Administrar recursos globales.

Resultados de transformaciones de MapForce

Altova MapForce realiza asignaciones de datos entre esquemas de entrada (ya existentes) y esquemas de salida (nuevos). Esta asignación, creada por el usuario de MapForce, se conoce como diseño de MapForce (o archivo MFD). Los archivos XML, de texto, bases de datos, etc. correspondientes a los esquemas de entrada se pueden usar como orígenes de datos. MapForce genera archivos de datos de resultados que se corresponden con el esquema de salida. Este documento de salida es el *resultado de una transformación de MapForce* que se convertirá en recurso global.

Si desea definir un archivo de datos generado con MapForce como recurso global, seleccione la opción *Resultado de una transformación de MapForce* en este cuadro de diálogo:

© 2015 Altova GmbH



- ***Un archivo de diseño de MapForce (archivo `.mfd`).*** Es necesario especificar este archivo en el cuadro de texto *El recurso apuntará al resultado de* del cuadro de diálogo (*ver imagen anterior*).
- ***Uno o más archivos de datos de entrada.*** El archivo MFD seleccionado se analiza y, en función de la información que contenga sobre los esquemas de entrada, el panel *Entradas* muestra una lista de los archivos de datos de entrada (*ver imagen anterior*). Si prefiere usar otro archivo, haga clic en el icono **Examinar** y seleccione un archivo diferente.
- ***Un archivo de resultados.*** Si el archivo MFD tiene varios esquemas de salida, estos aparecen en la lista del panel *Resultados* (*imagen anterior*) y es necesario seleccionar uno. Si la ubicación del archivo de resultados de un esquema de salida está definida en el archivo MFD, esta ubicación aparecerá en el panel *Resultados*. Por ejemplo, en la imagen anterior puede ver que el archivo MFD dice que el esquema de salida `Customers` tiene un archivo de datos XML predeterminado (llamado `CustomersOut.xml`), mientras que el esquema de salida `Text file` no tiene ninguna asociación de archivo en el archivo MFD. Puede usar la ubicación de archivo predeterminada que aparece en el panel *Resultados* o especificar una ubicación nueva. El resultado de la transformación de MapForce se guardará en la ubicación de archivo del esquema de salida seleccionado.

Este es el archivo que se usará como recurso global.

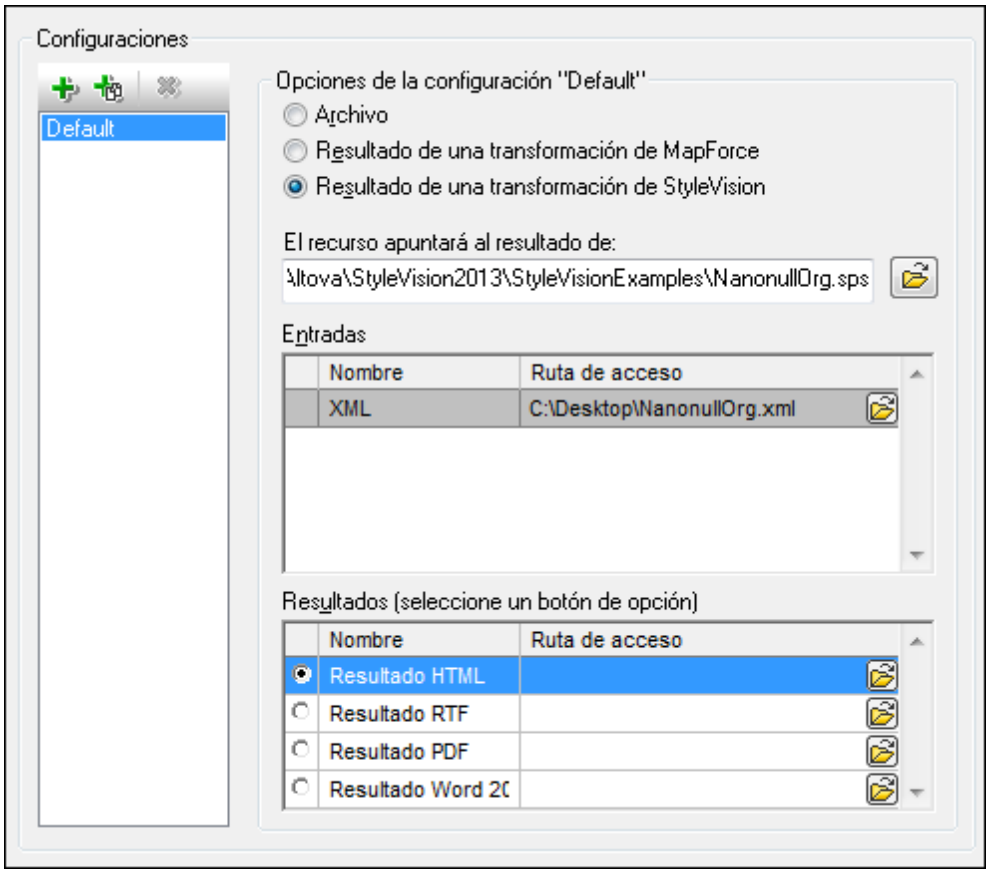
Nota: la ventaja de usar esta opción (resultado de una transformación de MapForce) es que la transformación se ejecuta al invocarse el recurso global. Esto significa que el recurso global contendrá los datos más recientes (de los archivos de entrada).

Nota: ya que para ejecutar la transformación se utiliza MapForce, es necesario tener instalado este producto de Altova. De lo contrario esta característica no funcionará.

Resultados de transformaciones de StyleVision

Altova StyleVision se utiliza para crear archivos SPS (StyleVision Power Stylesheet). Estos archivos generan hojas de estilos XSLT que se usan para transformar documentos XML en documentos de salida en varios formatos (HTML, PDF, RTF, Word 2007+, etc.) Si selecciona la opción *Resultado de una transformación de StyleVision*, el documento de resultados creado por StyleVision será el recurso global asociado a la configuración seleccionada.

Si selecciona la opción Transformación de StyleVision (*imagen siguiente*), es necesario especificar también estos archivos:



- ***Un archivo `.sps` (SPS).*** Seleccione el archivo SPS en el cuadro de texto *El recurso apuntará al resultado de* (*ver imagen anterior*).
- ***Archivos de entrada***. El archivo de entrada puede venir dado por el archivo SPS. Si es así, el archivo de entrada aparecerá automáticamente en el panel *Inputs*. En este panel

 © 2015 Altova GmbH

puede elegir el archivo de entrada que quiera. Si el panel está vacío, entonces añada un archivo.

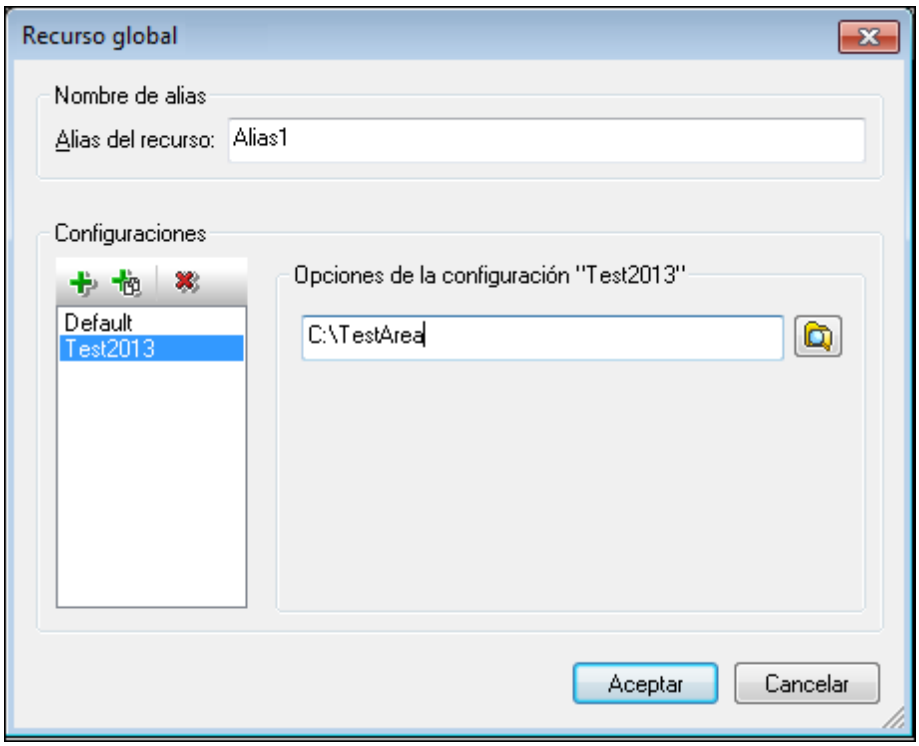
- ***Archivos de resultados.*** Seleccione el formato de salida en el panel *Resultados* y especifique la ubicación del archivo.

Nota: la ventaja de seleccionar esta opción (resultado de la transformación de StyleVision) es que la transformación se ejecuta al invocarse el recurso global. Esto significa que el recurso global contendrá los datos más recientes (de los archivos de entrada).

Nota: ya que para ejecutar la transformación se utiliza StyleVision, es necesario tener instalado este producto de Altova. De lo contrario esta característica no funcionará.

6.1.2 Carpetas

En el cuadro de diálogo "Recurso global" para carpetas (*imagen siguiente*), siga los pasos que aparecen más abajo para agregar un recurso de carpeta.



Iconos del cuadro de diálogo de recursos globales

Agregar configuración: abre el cuadro de diálogo "Agregar configuración", donde puede introducir el nombre de la configuración que se debe agregar.

Agregar una copia de configuración: abre el cuadro de diálogo "Agregar configuración", donde puede escribir el nombre de la configuración que debe crearse como copia de la configuración seleccionada.

Eliminar: elimina la configuración seleccionada.

Abrir: examina el sistema de archivos para buscar el archivo que se debe usar para crear el recurso global.

Definir el alias

Siga estos pasos para definir el alias (su nombre y su configuración):

1. *Póngale un nombre al alias:* escriba un nombre de alias en el cuadro de texto *Alias del recurso*.
2. *Añada configuraciones:* el panel Configuraciones, situado a la izquierda, tendrá una configuración predeterminada llamada *Default* (*imagen anterior*). Esta configuración

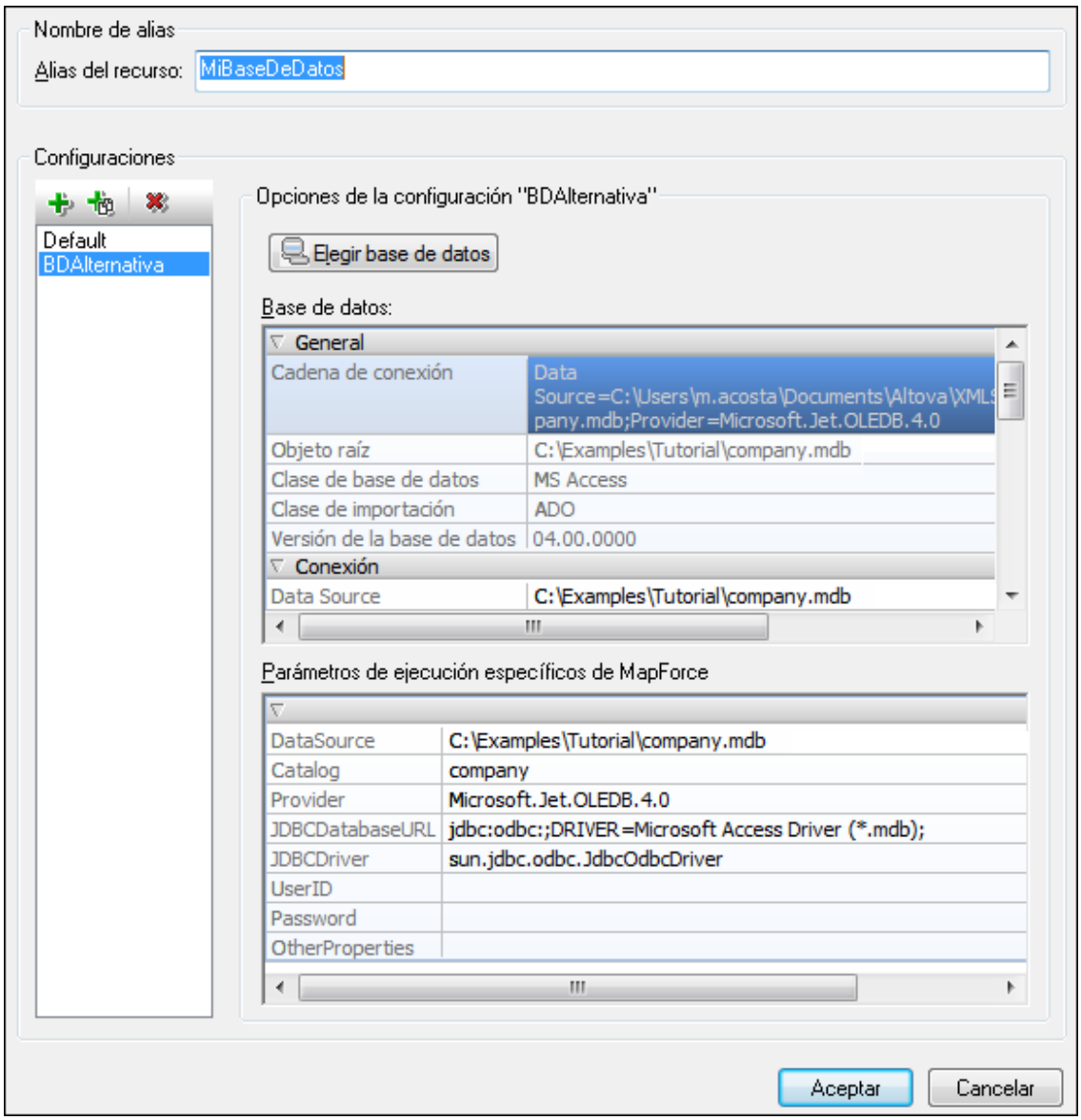
© 2015 Altova GmbH

Default no se puede eliminar ni cambiar de nombre. Puede introducir todas las configuraciones que quiera para el alias seleccionado. Para agregar una configuración, haga clic en el icono **Agregar configuración** o en el icono **Agregar copia de configuración**. Aparece el cuadro de diálogo "Agregar configuración". Escriba el nombre de la configuración y haga clic en **Aceptar**. La configuración nueva aparece en la lista del panel Configuraciones. Repita estos pasos para crear más configuraciones.

3. *Seleccione una carpeta como recurso de la configuración:* seleccione una de las configuraciones del panel Configuraciones y busque la carpeta que desea crear como recurso global.
4. *Si es necesario, defina más configuraciones*: indique un recurso de carpeta por cada configuración (es decir, repita el paso 3 tantas veces como configuraciones). Puede agregar configuraciones nuevas a la definición de alias en todo momento.
5. *Guarde la definición de alias:* en el cuadro de diálogo "Recurso global" haga clic en el botón **Aceptar** para guardar como recurso global el alias y todas sus configuraciones. El recurso global aparecerá en el cuadro de diálogo "Gestionar Recursos Globales", bajo el nodo *Carpetas*.

6.1.3 Bases de datos

En el cuadro de diálogo "Recurso global" para bases de datos (*imagen siguiente*), siga los pasos que aparecen más abajo para agregar un recurso de base de datos.



Iconos del cuadro de diálogo de recursos globales

Agregar configuración: abre el cuadro de diálogo "Agregar configuración", donde puede introducir el nombre de la configuración que se debe agregar.

Agregar una copia de configuración: abre el cuadro de diálogo "Agregar configuración", donde puede escribir el nombre de la configuración que debe crearse como copia de la configuración seleccionada.

Eliminar: elimina la configuración seleccionada.

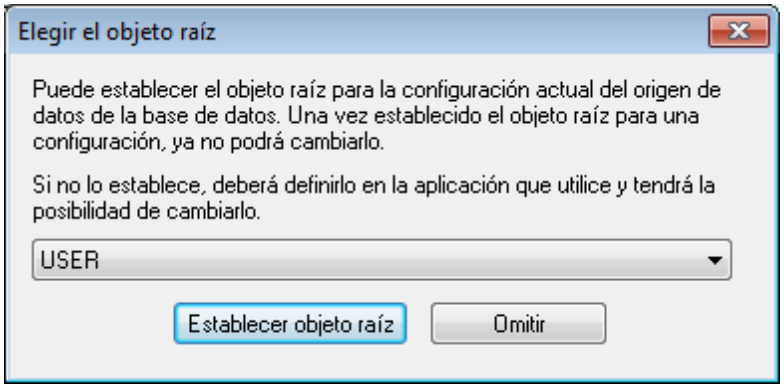
© 2015 Altova GmbH

Abrir: examina el sistema de archivos para buscar el archivo que se debe usar para crear el recurso global.

Definir el alias

Siga estos pasos para definir el alias (su nombre y su configuración):

1. *Póngale un nombre al alias:* escriba el nombre del alias en el cuadro de texto *Alias del recurso*.
2. *Agregar configuraciones:* el panel Configuraciones, situado a la izquierda, tendrá una configuración predeterminada llamada *Default* (*imagen anterior*). Esta configuración Default no se puede eliminar ni cambiar de nombre. Puede introducir todas las configuraciones que quiera para el alias seleccionado. Para agregar una configuración, haga clic en el icono **Agregar configuración** o en el icono **Agregar copia de configuración**. Aparece el cuadro de diálogo "Agregar configuración". Escriba el nombre de la configuración y haga clic en **Aceptar**. La configuración nueva aparece en la lista del panel Configuraciones. Repita estos pasos para crear las configuraciones que necesite.
3. *Inicie la selección de una base de datos como recurso de la configuración:* seleccione una de las configuraciones de panel Configuraciones y pulse el botón **Elegir base de datos**. Aparece el cuadro de diálogo "Crear conexión para recursos globales".
4. *Conectarse a la base de datos:* Seleccione si desea crear una conexión a la base de datos usando el asistente de conexión, una conexión existente, una conexión ADO, una conexión ODBC o JDBC.
5. *Seleccione el objeto raíz:* si se conecta a un servidor de base de datos en el que se puede seleccionar un objeto raíz, entonces aparece el cuadro de diálogo "Elegir el objeto raíz" (*imagen siguiente*). En este diálogo puede seleccionar el objeto raíz del servidor, que es el objeto raíz que se cargará cuando se utilice esta configuración. Para terminar haga clic en **Establecer objeto raíz**.


Elegir el objeto raíz
Puede establecer el objeto raíz para la configuración actual del origen de datos de la base de datos. Una vez establecido el objeto raíz para una configuración, ya no podrá cambiarlo.
Si no lo establece, deberá definirlo en la aplicación que utilice y tendrá la posibilidad de cambiarlo.
USER
Establecer objeto raíz
Omitir

   Si prefiere no seleccionar ningún objeto raíz (haciendo clic en el botón **Omitir**), podrá seleccionar el objeto raíz en el momento en que se cargue el recurso global.
6. *Si es necesario, defina más configuraciones:* especifique un recurso de base de datos por cada configuración que necesite (es decir, repita los pasos 3, 4 y 5 tantas veces como configuraciones necesite). Puede añadir una configuración nueva a la definición de alias en todo momento.
7. *Guarde la definición de alias:* en el cuadro de diálogo "Recurso global" haga clic en **Aceptar** para guardar como recurso global el alias y todas sus configuraciones. El recurso global aparecerá en el cuadro de diálogo "Administrar recursos globales".

6.2 Usar recursos globales

Hay varios tipos de recursos globales (de tipo archivo, carpeta y base de datos). Algunos casos de uso de los recursos globales en Authentic Desktop se describen aquí: Archivos y carpetas.

Selecciones que determinan el recurso que se utiliza

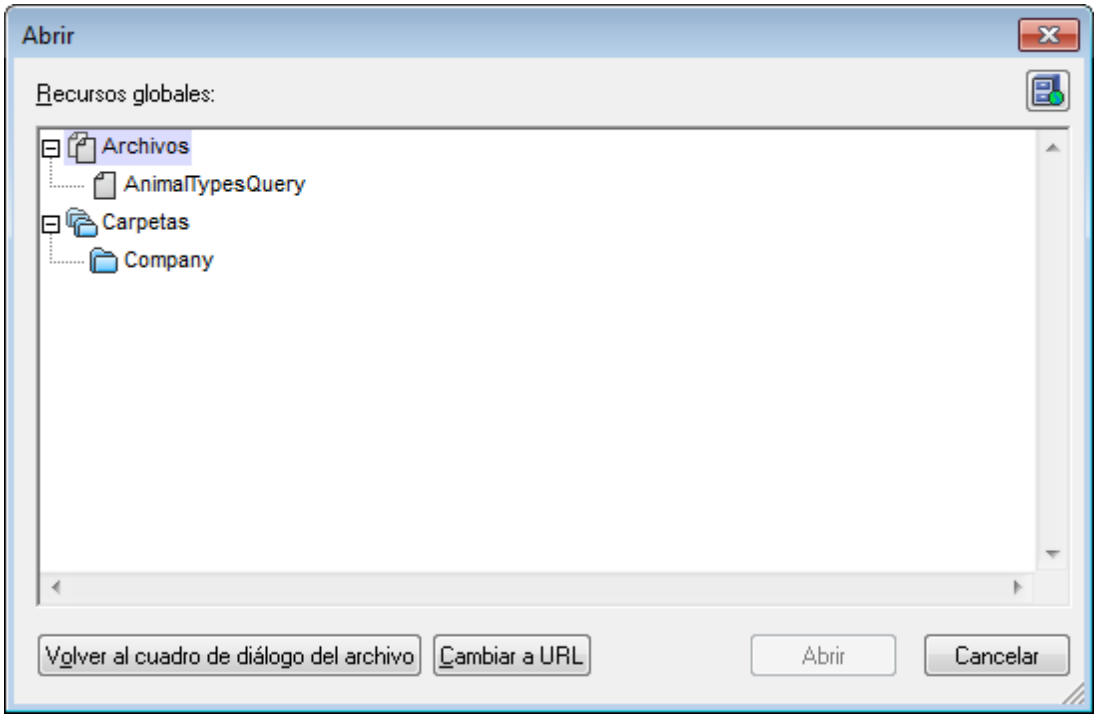
Hay dos selecciones generales que determinan qué recursos globales se pueden usar en general y cuáles se pueden usar en cada momento:

- El *archivo XML de recursos globales* seleccionado en el cuadro de diálogo Recurso global. Las definiciones de recursos globales presentes en el archivo XML de recursos globales están a disposición de todos los archivos que estén abiertos en la aplicación. Las únicas definiciones disponibles son las del archivo XML de recursos globales. Puede cambiar de archivo XML de recursos globales cuando quiera y las definiciones del nuevo archivo activo sustituyen inmediatamente a las del archivo activo anterior. Por tanto, el archivo XML de recursos globales activo determina: (i) qué recursos globales se pueden asignar y (ii) qué recursos globales se pueden buscar (p. ej. si se asigna un recurso global de un archivo XML de recursos globales, pero en el archivo XML de recursos globales activo no hay ningún recurso global que se llame así, entonces no se puede buscar el recurso global, o alias, asignado).
- La *configuración activa* seleccionada en la opción de menú **Herramientas | Configuración activa** o en la barra de herramientas Recursos globales. Al hacer clic en este comando (o en la lista desplegable de la barra de herramientas) aparecen las configuraciones de todos los alias. Si selecciona una configuración, la configuración es la configuración activa para toda la aplicación. Es decir, cuando se usa un recurso global (o alias), se carga el recurso correspondiente a la configuración activa del alias usado. La configuración activa se aplica a todos los alias usados. Si un alias no tiene una configuración que se llame como la configuración activa, entonces se usa la configuración predeterminada de ese alias. La configuración activa no es relevante a la hora de asignar recursos. Sólo es importante a la hora de utilizar los recursos.

© 2015 Altova GmbH

6.2.1 Asignar archivos y carpetas

En este apartado explicamos cómo se asignan recursos globales de tipo archivo y carpeta. Los recursos globales de tipo archivo no se asignan igual que los recursos globales de tipo carpeta. En los casos de uso que damos como ejemplo y que aparecen en este apartado, al hacer clic en el botón **Cambiar a recursos globales** aparece el cuadro de diálogo "Abrir recurso global" (*imagen siguiente*).



Administrar recursos globales: abre el cuadro de diálogo Administrar recursos globales.

Si selecciona un archivo en este cuadro de diálogo, se asigna el archivo seleccionado. Si selecciona una carpeta, aparece un cuadro de diálogo "Abrir" donde puede examinar la carpeta para buscar el archivo deseado. La ruta de acceso al archivo seleccionado es relativa al recurso de carpeta. Por tanto, si un recurso global de tipo carpeta tuviese dos configuraciones, cada una apuntando a una carpeta diferente, se podrían seleccionar dos archivos con el mismo nombre pero ubicados en carpetas diferentes. Esto puede ser muy útil durante la fase de pruebas, por ejemplo.

Puede cambiar al cuadro de diálogo del archivo o al de la URL con solo pulsar el botón correspondiente. El icono **Administrar recursos globales** situado en la esquina superior derecha abre el cuadro de diálogo Administrar recursos globales.

Casos de uso

Puede usar recursos globales de tipo archivo y carpeta en estos casos:

- Al abrir recursos globales
- Al guardar un archivo como recurso global
- En transformaciones XSLT y ejecuciones de XQuery

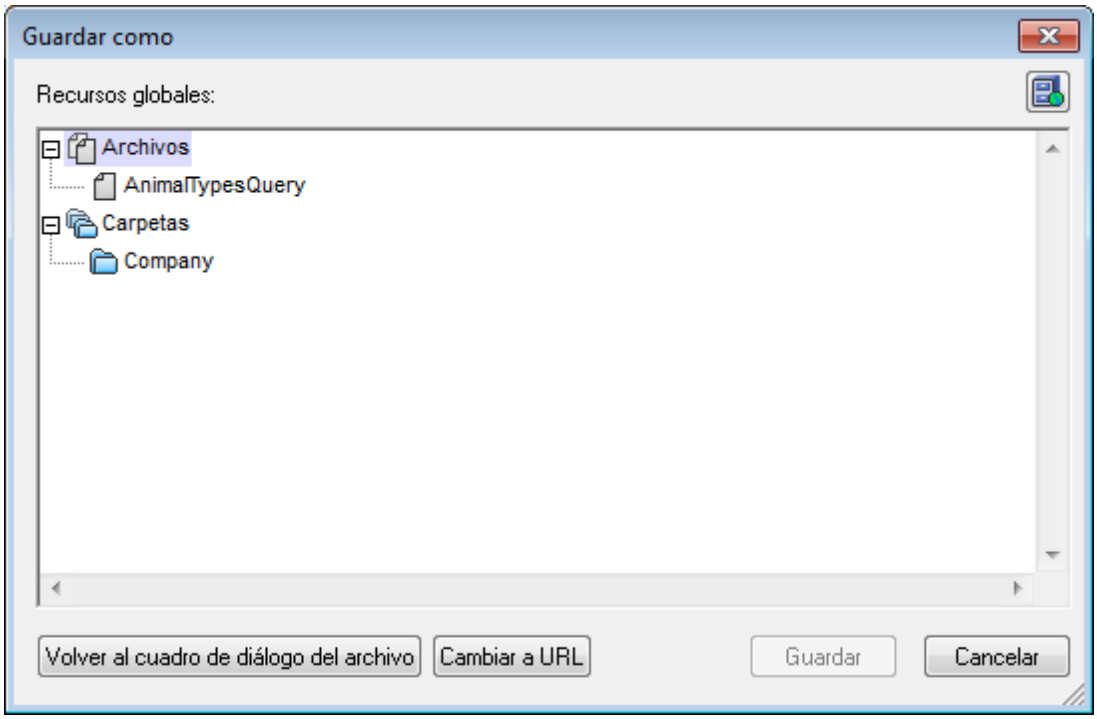
- Al asignar una SPS

Abrir recursos globales

Puede abrir recursos globales en Authentic Desktop con el comando **Archivo | Abrir (Cambiar a recursos globales)**. Si se trata de un recurso global de tipo archivo, el archivo se abre directamente. Si se trata de un recurso global de tipo carpeta, entonces aparece un cuadro de diálogo "Abrir" donde puede examinar la carpeta seleccionada y buscar el archivo deseado. Una ventaja de editar archivos a través de recursos globales es que los archivos relacionados se pueden guardar en configuraciones de un recurso global y para abrirlos basta con cambiar de configuración. Recuerde guardar los cambios realizados en el archivo antes de cambiar de configuración.

Guardar un archivo como recurso global

Los archivos recién creados se pueden guardar como recurso global. También puede abrir un archivo ya existente y guardarlo como recurso global. Al hacer clic en los comandos **Archivo | Guardar** o **Archivo | Guardar como** aparece el cuadro de diálogo "Guardar". Pulse el botón **Cambiar a recursos globales** para acceder a los recursos globales disponibles (*imagen siguiente*), es decir, los alias definidos en el archivo XML de recursos globales actual.



Seleccione un alias y pulse el botón **Guardar**. Si el alias es un alias de archivo el archivo se guarda directamente. Si el alias es un alias de carpeta, entonces aparece un cuadro de diálogo donde debe escribir el nombre con el que desea guardar el archivo. En ambos casos, el archivo se guarda en la ubicación que se definió para la configuración activa actual.

Nota: cada configuración apunta a una ubicación de archivo concreta, que se indica en la definición de la configuración. Si el archivo que desea guardar como recurso global no tiene la misma extensión de archivo que el archivo guardado en la ubicación de archivo de la configuración, entonces puede haber errores de edición y validación al abrir este

© 2015 Altova GmbH

recurso global en Authentic Desktop. El motivo es que, al abrir el archivo, Authentic Desktop asume el tipo de archivo indicado en la definición de la configuración.

Transformaciones XSLT

Al hacer clic en los comandos **XSL/XQuery | Transformación XSL** o **XSL/XQuery | Transformación XSL-FO** aparece un cuadro de diálogo en el que puede buscar el archivo XSLT, XQuery y XML pertinentes. Haga clic en el botón **Examinar** y después en el botón **Cambiar a recursos globales**. Aparece el cuadro de diálogo "Abrir" (*ver primera imagen de este apartado*). Para la transformación se usa el archivo asociado a la configuración activa del recurso global seleccionado en este diálogo.

Asignar una SPS

Cuando asigne una hoja de estilos de StyleVision a un archivo XML (**Authentic | Asignar hoja de estilos de StyleVision**), puede usar un recurso global para localizar la hoja de estilos. Haga clic en el botón **Examinar** y después en el botón **Cambiar a recursos globales** y aparece el diálogo "Abrir recurso global" (*ver primera imagen de este apartado*). Si se selecciona un recurso global, puede cambiar el documento XML de la vista Authentic con solo cambiar la configuración activa en Authentic Desktop.

6.2.2 Cambiar de configuración

En la aplicación no puede haber más de una configuración activa al mismo tiempo y la configuración activa afecta a toda la aplicación. Esto quiere decir que la configuración activa está activa para todos los alias en todos los archivos que estén abiertos en la aplicación y en todas las conexiones a orígenes de datos. Si un alias no tiene una configuración con el nombre de la configuración activa, entonces se usa la configuración predeterminada (*Default*) de dicho alias.

Hay dos maneras de cambiar de configuración activa:

- Con el comando de menú **Herramientas | Configuración Activa**, donde puede seleccionar qué configuración que desea usar.
- En el cuadro combinado de la barra de herramientas Recursos globales (*imagen siguiente*), donde puede seleccionar la configuración deseada.

De este manera, al cambiar de configuración activa, puede cambiar los archivos de origen que se asignaron a través de recursos globales.

© 2015 Altova GmbH

7 Control de código fuente

La compatibilidad del control de código fuente en el Authentic Desktop se encuentra disponible a través de la API del complemento del control de código fuente de Microsoft (antes conocido como MSSCCI API), versiones 1.1, 1.2 y 1.3. Este le permite ejecutar los comandos del control del código fuente tales como "Check-in" o "Check-out" directamente desde el Authentic Desktop a cualquier sistema de control de código fuente virtual que les permite conectarse a los nativos clientes o a las terceras partes mediante la API del complemento del control de código fuente de Microsoft.

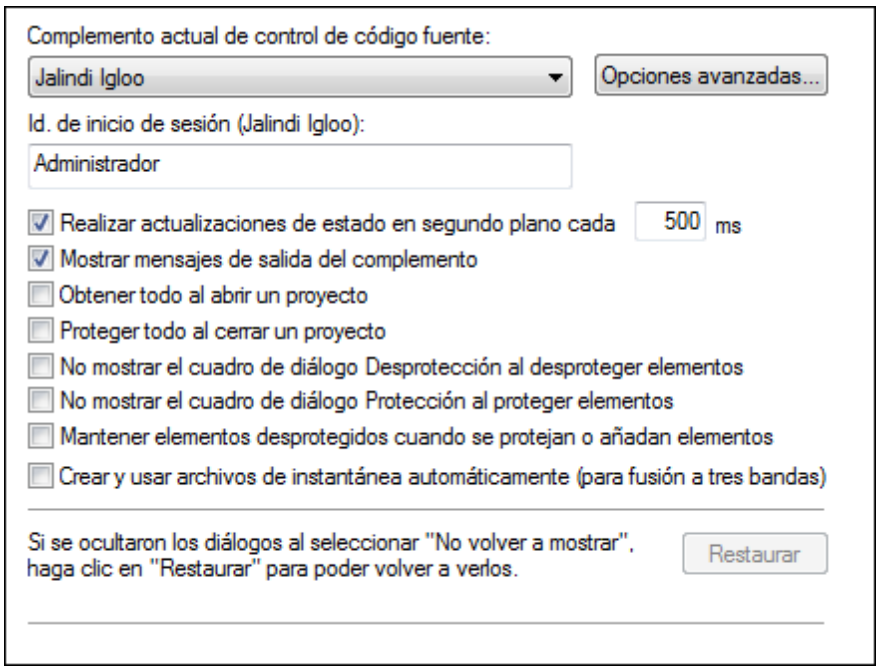
Puede usar como proveedor del elemento de control fuente cualquier complemento comercial o no comercial que sea compatible con la API del elemento de control de Microsoft y puede conectarse a un sistema de control de versión compatible. Consulte los Sistemas de control de código fuente para ver el listado de los sistemas de control del código fuente y los complementos que se han evaluado por Altova.

Instalar y configurar el proveedor el proveedor del control del código fuente

Para ver los proveedores del control del código fuente que aparecen disponibles en su sistema, realice los siguientes pasos:

1. Haga clic en **Opciones** del menú **Herramientas**.
2. Haga clic sobre la pestaña **Control de código fuente**.

Cualquier complemento de control de código fuente que sea compatible con la API del complemento de control del código fuente de Microsoft se visualiza en la lista desplegable del **complemento de control de código fuente actual**


Complemento actual de control de código fuente:
Jalindi Igloo
Opciones avanzadas...
Id. de inicio de sesión (Jalindi Igloo):
Administrador
Realizar actualizaciones de estado en segundo plano cada
500
ms
Mostrar mensajes de salida del complemento
Obtener todo al abrir un proyecto
Proteger todo al cerrar un proyecto
No mostrar el cuadro de diálogo Desprotección al desproteger elementos
No mostrar el cuadro de diálogo Protección al proteger elementos
Mantener elementos desprotegidos cuando se protejan o añadan elementos
Crear y usar archivos de instantánea automáticamente (para fusión a tres bandas)
Si se ocultaron los diálogos al seleccionar "No volver a mostrar", haga clic en "Restaurar" para poder volver a verlos.
Restaurar

Si no se puede encontrar un elemento compatible en su sistema, le aparecerá el siguiente mensaje:

```
"El registro de los proveedores del elemento de control del código fuente
instalado no se pudo encontrar o está incompleto."
```

Algunos sistemas de control de código fuente podrían no instalar el control de código fuente de forma automática, en este caso necesitará instalarlo de forma separada. Para obtener más instrucciones, diríjase a los manuales del sistema de control de código fuente correspondientes. Bajo la siguiente entrada de registro de su sistema operativo, se registrará un complemento (proveedor) compatible con la API del complemento del control del código de fuente de Microsoft:

```
HKEY_LOCAL_MACHINE\SOFTWARE\SourceCodeControlProvider\InstalledSCCProviders
```

En el momento de realizar la instalación correcta, los complementos estarán disponibles de forma automática en el listado de complementos que aparece disponible en Authentic Desktop.

Acceso a los comandos de control del código fuente

Los comandos relacionados con el control de código fuente se encuentran disponibles en el menú **Proyecto | Control de código fuente**.

Problemas relacionados con los recursos y con la velocidad

Las bases de datos de código fuente de gran tamaño pueden introducir una sanción de velocidad/ recurso cuando se llevan a cabo actualizaciones en segundo plano de forma automática.

Puede incrementar la velocidad de su sistema desactivando (o incrementando el **intervalo de)** la opción **Llevar a cabo actualizaciones en segundo plano cada... segundos** que se localiza en la pestaña "Control de código fuente" al que se accede mediante los comandos **Herramientas | Opciones**.

Nota: La versión de **64 bits** de su aplicación Altova es compatible de forma automática con cualquiera de los programas de control de código fuente de 32 bits, la opción **Llevar a cabo actualizaciones en segundo plano cada... segundos** aparece marcada en gris de forma automática y no se puede seleccionar.

Diferencias con el DiffDog de Altova

Puede configurar muchos sistemas de control de códifo fuente (incluyendo Git y TortoiseSVN) de tal forma que usan Altova DiffDig como una herramienta diferenciadora. Para obtener más información acerca de DiffDog, consulte http://www.altova.com/diffdog.html. Para obtener más información sobre DiffDog, consulte http://www.altova.com/documentation.html.

© 2015 Altova GmbH

7.1 Configurar el control de código fuente

El mecanismo para configurar el control de código fuente y poner archivos de un proyecto de Authentic Desktop bajo control de código fuente se describe a continuación:

1. Si esto aún no se hubiera realizado, instale el sistema de control de código fuente (consulte Sistemas de control de código fuente compatibles) y configure la base de datos del control de código fuente (repositorio) en el que desea guardar su trabajo.
2. Cree una carpeta local de la copia de trabajo donde se guardarán los archivos de trabajo que desea poner bajo control de código fuente. La carpeta que contiene todas sus carpetas y archivos de la copia de trabajo se llama *carpeta local*. La ruta de acceso de la carpeta local se denomina *ruta de acceso local*. La carpeta se enlazará a una carpeta determinada del repositorio.
3. En su aplicación de Altova cree una carpeta de proyecto de la aplicación donde debe añadir los archivos que desea poner bajo control de código fuente. El proyecto de la aplicación es una organización abstracta de los archivos. Los archivos del proyecto hacen referencia a archivos físicos guardados en la carpeta local..
4. En la BD del sistema de control de código fuente (conocida también como repositorio) se crea una carpeta que está enlazada a la carpeta local. Esta carpeta (denominada *carpeta enlazada*) es una réplica de la estructura de la carpeta local. De este modo los archivos que se ponen bajo control de código fuente se colocan en el puesto correcto dentro de la jerarquía de archivos. La carpeta enlazada se suele crear cuando se agrega por primera vez un archivo o un proyecto de la aplicación al control de código fuente. Para más información sobre la estructura de carpetas del repositorio, consulte el apartado Proyecto de la aplicación.
5. Los archivos del proyecto se añaden al control de código fuente con el comando **Proyecto | Control de código fuente | Agregar al control de código fuente**. Cuando se agrega por primera vez un proyecto o un archivo del proyecto por primera vez, se crea la carpeta enlazada.
6. Las operaciones de control de código fuente, como la protección o desprotección de archivos, se realizan con los comandos del submenú **Proyecto | Control de código fuente**. Estos comandos se describen en el apartado Menú Proyecto de la *Referencia del usuario*.

Nota: si desea cambiar de proveedor de control de código fuente, puede hacerlo de dos maneras diferentes: (i) en el cuadro de diálogo "Opciones" (**Herramientas | Opciones | Control de código fuente**) o (ii) en el cuadro de diálogo "Cambiar control de código fuente" (**Proyecto | Control de código fuente | Cambiar control de código fuente**).

7.2 Sistemas de control de código fuente compatibles

La lista que aparece a continuación enumera los sistemas de control de código fuente compatibles con Authentic Desktop, junto con sus clientes de control de código fuente(SCCs). Los sistemas de control de código fuente de la lista aparecen ordenados alfabéticamente. Tenga en cuenta los siguiente:

- Altova ha implementado la API del complemento de control de código fuente de Microsoft (versiones 1.1, 1.2 y 1.3) en Authentic Desktop y ha evaluado la compatibilidad con los controladores de la lista y los sistema de control de revisión. Por tanto, es de esperar que el Authentic Desktop continuará siendo compatible con dichos productos siempre y cuando estén actualizados.
- Los clientes del control de código fuente no aparecen en la lista siguiente pero los que implementan la API del complemento de control de código fuente de Microsoft, también deberán funcionar con Authentic Desktop.

Sistema de control de código fuente	Clientes de control del código fuente
AccuRev 4.7.0 Windows	AccuBridge para Microsoft SCC 2008.2
Bazaar 1.9 Windows	Aigenta Unified SCC 1.0.6
Borland StarTeam 2008	Borland StarTeam Cross-Platform Client 2008 R2
Codice Software Plastic SCM Professional 2.7.127.10 (Server)	Codice Software Plastic SCM Professional 2.7.127.10 (SCC Plugin)
Collabnet Subversion 1.5.4	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 version 1.6.3.1 • TamTam SVN SCC 1.2.24
ComponentSoftware CS-RCS (PRO) 5.1	ComponentSoftware CS-RCS (PRO) 5.1
Dynamsoft SourceAnywhere for VSS 5.3.2 Standard/Professional Server	Dynamsoft SourceAnywhere for VSS 5.3.2 Client
Dynamsoft SourceAnywhere Hosted	Dynamsoft SourceAnywhere Hosted Client (22252)
SourceAnywhere Standalone 2.2 Server	Dynamsoft SourceAnywhere Standalone 2.2 Client
IBM Rational ClearCase 7.0.1 (LT)	IBM Rational ClearCase 7.0.1 (LT)
March-Hare CVSNT 2.5 (2.5.03.2382)	Aigenta Unified SCC 1.0.6
March-Hare CVS Suite 2008	• Jalindi Igloo 1.0.3 • March-Hare CVS Suite Client 2008 (3321) • PushOK CVS SCC NT 2.1.2.5 • PushOK CVS SCC x64 version 2.2.0.4 • TamTam CVS SCC 1.2.40
Mercurial 1.0.2 for Windows	Sergey Antonov HgSCC 1.0.1
Microsoft SourceSafe 2005 with CTP	Microsoft SourceSafe 2005 with CTP

© 2015 Altova GmbH

Sistema de control de código fuente	Clientes de control del código fuente
Microsoft Visual Studio Team System 2008/2010 Team Foundation Server	Microsoft Team Foundation Server 2008/2010 MSSCCI Provider
Perforce 2008 P4S 2008.1	Perforce P4V 2008.1
PureCM Server 2008/3a	PureCM Client 2008/3a
QSC Team Coherence Server 7.2.1.35	QSC Team Coherence Client 7.2.1.35
Reliable Software Code Co-Op 5.1a	Reliable Software Code Co-Op 5.1a
Seapine Surround SCM Client/Server for Windows 2009.0.0	Seapine Surround SCM Client 2009.0.0
Serena Dimensions Express/CM 10.1.3 for Win32 Server	Serena Dimensions 10.1.3 for Win32 Client
Softimage Alienbrain Server 8.1.0.7300	Softimage Alienbrain Essentials/Advanced Client 8.1.0.7300
SourceGear Fortress 1.1.4 Server	SourceGear Fortress 1.1.4 Client
SourceGear SourceOffsite Server 4.2.0	SourceGear SourceOffsite Client 4.2.0 (Windows)
SourceGear Vault 4.1.4 Server	SourceGear Vault 4.1.4 Client
VisualSVN Server 1.6	• Aigenta Unified SCC 1.0.6 • PushOK SVN SCC 1.5.1.1 • PushOK SVN SCC x64 version 1.6.3.1 • TamTam SVN SCC 1.2.24

7.3 Carpeta de trabajo local

Los archivos con los que trabajará se deberían guardar en una jerarquía dentro de la carpeta local de la copia de trabajo:

```
Carpeta local de la copia de trabajo
|
|-- MiProyecto.spp
|-- QuickStart
|   |-- QuickStart.css
|   |-- QuickStart.xml
|   |-- QuickStart.xsd
|-- Grouping
|   | -- Persons
|   |    | -- Persons.xml
```

El archivo de proyecto de la aplicación (archivo `.spp`) está por lo general dentro de la carpeta local de la copia de trabajo (*ver ejemplo anterior*).

Cuando uno de los archivos de esta carpeta (copia de trabajo) se ponen bajo control de código fuente, la estructura de la carpeta local de la copia de trabajo se reproduce por completo o parcialmente en el repositorio. Por ejemplo, si el archivo `Persons.xml` de la carpeta local del ejemplo anterior se pone bajo control de código fuente, entonces su ruta de acceso en el repositorio sería esta:

```
[CarpetaRepositorio]/MiProyecto/Grouping/Persons/Persons.xml
```

La carpeta `MiProyecto` de la carpeta del repositorio está enlazada a la carpeta local. Por lo general tendrá el nombre del proyecto, pero puede ponerle otro nombre si así lo desea.

Si se pone bajo control de código fuente todo el proyecto de la aplicación (seleccione el nombre del proyecto en la ventana Proyecto y póngalo bajo control de código fuente), entonces se recrea toda la estructura de la carpeta local en el repositorio.

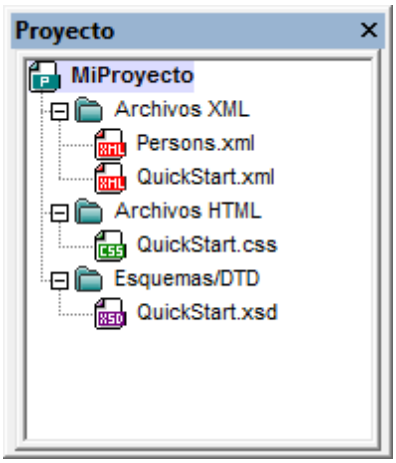
Nota: los archivos que están fuera de la carpeta local de la copia de trabajo se pueden añadir al proyecto de la aplicación. Pero dependiendo del sistema SCS que utilice podrá poner esos archivos bajo control de código fuente o no. Algunos sistemas de control de código fuente tienen problemas a la hora de poner en el repositorio archivos situados fuera de la carpeta local. Por tanto, recomendamos que guarde en la carpeta local todos los archivos que quiera poner bajo control de código fuente.

© 2015 Altova GmbH

7.4 Proyecto de la aplicación

Cree o cargue el proyecto de su aplicación de Altova que quiere poner bajo control de código fuente. Si desea poner un solo archivo bajo control de código fuente, este archivo debe estar incluido en el proyecto (porque al control de código fuente solamente se puede acceder desde un proyecto).

Por ejemplo, imagine que trabaja con un proyecto de la aplicación XMLSpy. Las propiedades del proyecto están guardadas en un archivo `.spp`. En la aplicación, el proyecto se muestra en la ventana Proyecto (*imagen siguiente*). El proyecto de la imagen se llama `MiProyecto` y sus propiedades están guardadas en el archivo `MiProyecto.spp`.



Puede poner bajo control de código fuente el proyecto entero (todos sus archivos) o solamente algunos archivos del proyecto. **Solamente se pueden poner bajo control de código fuente los archivos que están incluidos en el proyecto**. Por tanto, para poner archivos bajo control de código fuente, antes debe agregarlos al proyecto. El archivo de proyecto (archivo `.spp`) se pone bajo control de código fuente automáticamente cuando uno de sus archivos se ponen bajo control de código fuente.

El proyecto y sus archivos se ponen bajo control de código fuente con el comando **Proyecto | Control de código fuente | Agregar al control de código fuente** (*ver el apartado siguiente*).

No obstante, recuerde que la estructura de la carpeta del repositorio no se corresponde a la estructura de la carpeta del proyecto (*imagen anterior*) sino a la estructura de la carpeta local de la copia de trabajo (*ver más abajo*). En el ejemplo siguiente observe que la estructura de la carpeta `MiProyecto` del repositorio se corresponde con la de la carpeta local de la copia de trabajo. Observe también que la carpeta enlazada está dentro de la carpeta del repositorio.

```
Carpeta local de la copia de trabajo          Repositorio
|                                                 |
|-- MiProyecto.spp                                |-- MiProyecto (enlazada
a la copia local)
|-- QuickStart                                    |   |-- MiProyecto.spp
|   |-- QuickStart.css                            |   |-- QuickStart
|   |-- QuickStart.xml                            |   |   |--
QuickStart.css
|   |-- QuickStart.xsd                            |   |   |--
QuickStart.xml
```

```
|-- Grouping                                          |   |   |--
QuickStart.xsd
|   | -- Persons                                      |   |-- Grouping
|   |    | -- Persons.xml                             |   |   | -- Persons
                                                      |   |   |    | --
Persons.xml
```

Nota: un proyecto de la aplicación puede contener carpetas de proyecto (verdes) o carpetas externas (amarillas). Solamente puede poner bajo control de código fuente archivos que estén dentro de carpetas de proyecto (verdes). Los archivos de las carpetas externas no se pueden poner bajo control de código fuente.

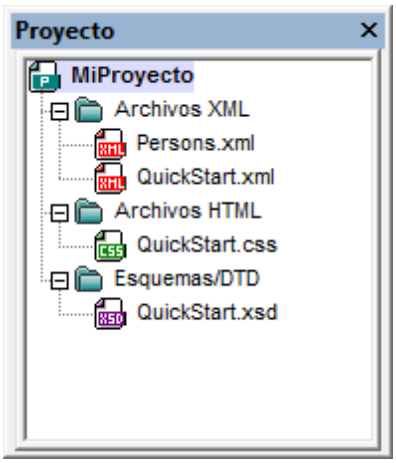
Nota: los archivos que están fuera de la carpeta local de la copia de trabajo se pueden añadir al proyecto de la aplicación. Pero dependiendo del sistema SCS que utilice podrá poner esos archivos bajo control de código fuente o no. Algunos sistemas de control de código fuente tienen problemas a la hora de poner en el repositorio archivos situados fuera de la carpeta local. Por tanto, recomendamos que guarde en la carpeta local todos los archivos que quiera poner bajo control de código fuente.

© 2015 Altova GmbH

7.5 Agregar al control de código fuente

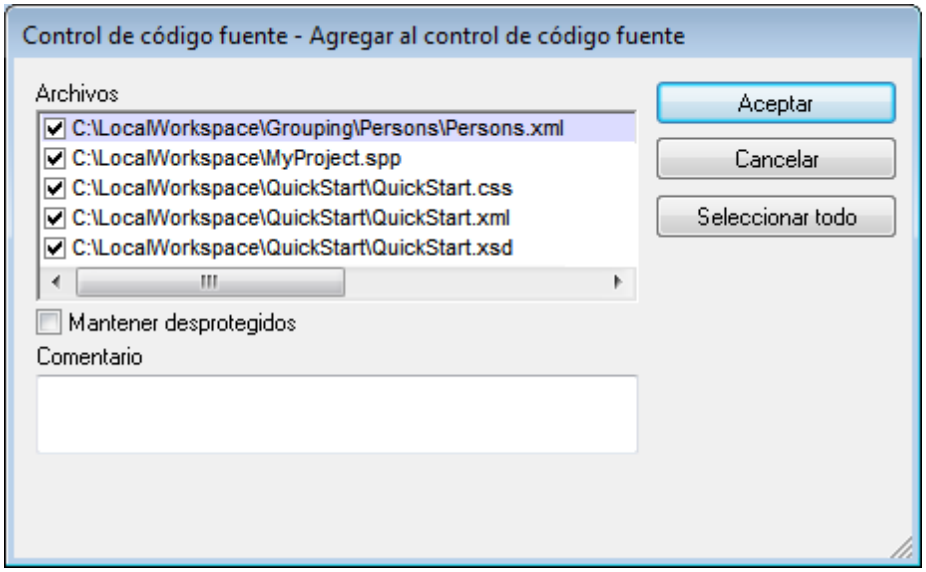
Cuando se agrega el proyecto al control de código fuente se crean automáticamente los enlaces correspondientes y la estructura del repositorio antes de poner el archivo de proyecto (archivo `.spp`) o los archivos bajo control de código fuente. A continuación explicamos cómo agregar el proyecto al control de código fuente.

Seleccione el proyecto en la ventana Proyecto (*imagen siguiente*) y asegúrese de que está resaltado el título del proyecto (como en la imagen). También puede seleccionar uno o varios archivos. Si añade un archivo al control de código fuente, se añade también el archivo de proyecto (archivo `.spp`).



Después seleccione el comando **Proyecto | Control de código fuente | Agregar al control de código fuente**. Aparecen los cuadros de diálogo de conexión y configuración de sus sistema de control de código fuente. (Para cambiar de sistema haga clic en **Proyecto | Control de código fuente | Cambiar control de código fuente**.)

Siga las instrucciones del sistema de control de código fuente para realizar la conexión y configurar las opciones. Cuando termine, los archivos que seleccionó y el archivo de proyecto (archivo .spp) aparecen en el cuadro de diálogo "Agregar al control de código fuente" (*imagen siguiente*). Seleccione los archivos que desea agregar y pulse el botón **Aceptar**.



Los archivos se añadirán al repositorio y estarán protegidos o desprotegidos dependiendo de si marcó o no la casilla *Mantener desprotegidos*.

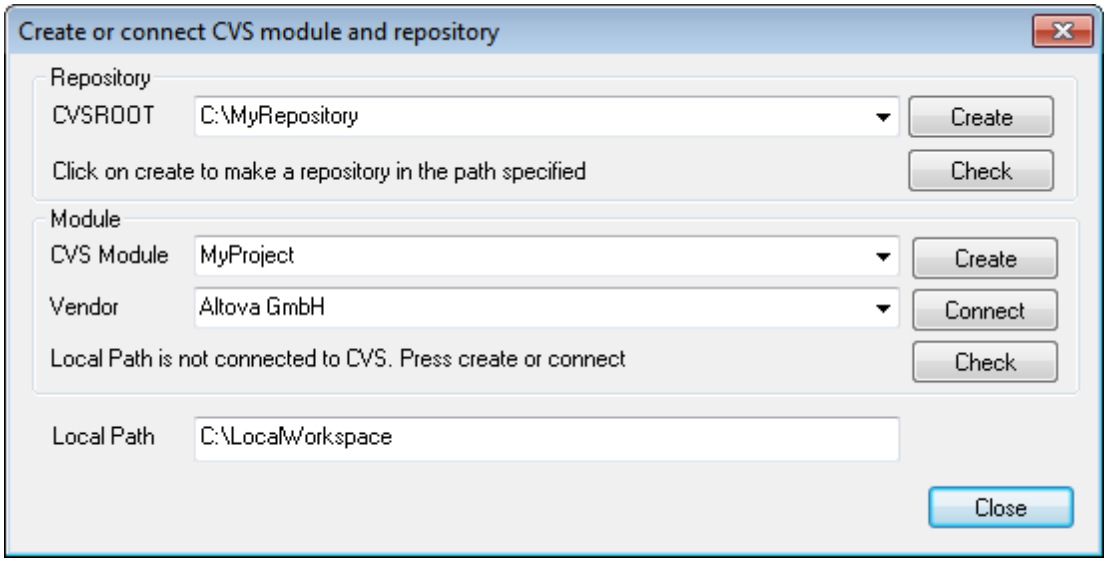
Notas de configuración

Puede que se solicite crear una carpeta para el proyecto en el repositorio (si todavía no se creó). La carpeta local de la copia de trabajo estará enlazada a esta carpeta del repositorio (*ver ejemplo siguiente)*.

```
Carpeta local de la copia de trabajo            Repositorio
|                                                |
|-- MiProyecto.spp                               |-- MiProyecto (enlazada
a la copia local)
|-- QuickStart                                   |   |-- MiProyecto.spp
|   |-- QuickStart.css                           |   |-- QuickStart
|   |-- QuickStart.xml                           |   |   |--
QuickStart.css
|   |-- QuickStart.xsd                           |   |   |--
QuickStart.xml
|-- Grouping                                     |   |   |--
QuickStart.xsd
|   | -- Persons                                 |   |-- Grouping
|   |    | -- Persons.xml                        |   |   | -- Persons
                                                 |   |   |    | --
Persons.xml
```

Por ejemplo, aquí puede ver el cuadro de diálogo de configuración de Jalindi Igloo. El campo *CVSROOT* es la ruta de acceso de la carpeta del repositorio.

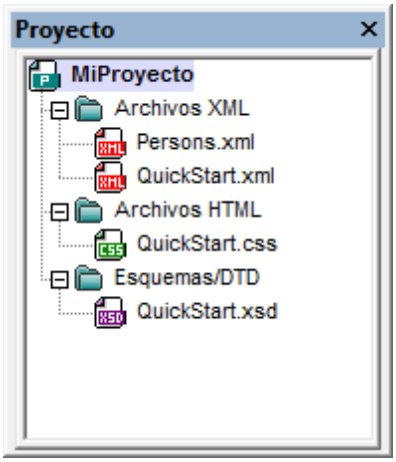
© 2015 Altova GmbH



En la imagen anterior, la ruta de acceso local se corresponde con la ruta de acceso del módulo CVS `MiProyecto`.

7.6 Trabajar con control de código fuente

Para trabajar con control de código fuente seleccione el proyecto, una carpeta de proyecto o un archivo del proyecto en la ventana Proyecto (imagen siguiente) y seleccione un comando del menú **Proyecto | Control de código fuente**. Los comandos **Proteger** y **Desproteger** también están disponibles en el menú contextual que aparece al hacer clic con el botón secundario en las entradas de la ventana Proyecto.



En esta sección describimos las características principales del control de código fuente:

- Agregar o quitar del control de código fuente
- Proteger, Desproteger
- Obtener archivos como archivos de solo lectura
- Copiar y compartir desde el control de código fuente
- Cambiar control de código fuente

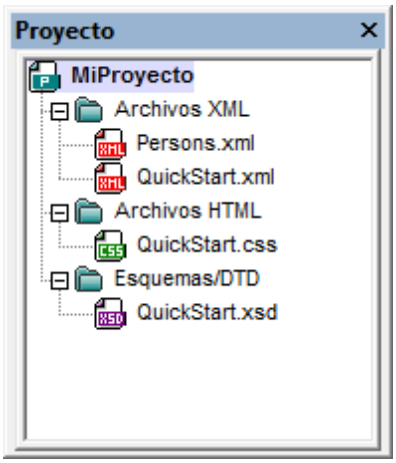
Los demás comandos del menú **Proyecto | Control de código fuente** se describen en la *Referencia del usuario*. Para más información consulte la documentación de sus sistema de control de código fuente.

© 2015 Altova GmbH

7.6.1 Agregar o quitar del control de código fuente

Agregar

Después de añadir un proyecto al control de código fuente, puede añadir archivos o grupos de archivos al sistema de control. Seleccione el archivo en la ventana Proyecto y haga clic en el comando **Proyecto | Control de código fuente | Agregar al control de código fuente**. Para seleccionar varios archivos mantenga pulsada la tecla **Ctrl** mientras hace clic en los archivos que desea añadir. Si ejecuta el comando en una carpeta de proyecto (de color verde), entonces se agregan al control de código fuente todos los archivos y subcarpetas de la carpeta.



Cuando se agregan archivos al control de código fuente, la jerarquía de la carpeta local se copia al repositorio (no la jerarquía de la carpeta de proyecto). Es decir, si un archivo está en una subcarpeta de la carpeta local, entonces se crean en el repositorio la carpeta primaria y las demás carpetas antecesoras del archivo.

Cuando se añade el primer archivo de un proyecto al control de código fuente, se crean los enlaces correspondientes en el repositorio y se añade automáticamente el archivo de proyecto (archivo `.spp`). Para más información consulte el apartado Agregar al control de código fuente.

Símbolos de control de código fuente

Las carpetas y los archivos aparecen con un icono diferente dependiendo de su estado:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Quitar

Para quitar un archivo del control de código fuente, selecciónelo y haga clic en el comando **Proyecto | Control de código fuente | Quitar del control de código fuente**. También puede quitar: (i) archivos de una carpeta de proyecto ejecutando el comando sobre la carpeta y (ii) todo el proyecto, ejecutando el comando sobre el proyecto.

7.6.2 Proteger, desproteger

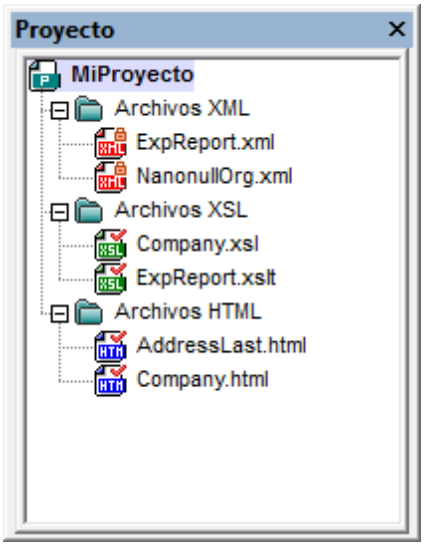
Tras poner un archivo de proyecto bajo control de código fuente, puede desprotegerlo o protegerlo. Para ello seleccione el archivo (en la ventana Proyecto) y seleccione el comando pertinente en el menú **Proyecto | Control de código fuente**, es decir **Desproteger** o **Proteger**.

Cuando se desprotege un archivo, se pone una copia del archivo del repositorio en la carpeta local. Los archivos desprotegidos se pueden editar. Si un archivo que está bajo control de código fuente no se desprotege, no se puede editar. Después de editar el archivo, los cambios se pueden guardar en el repositorio protegiendo el archivo. Incluso si no guarda el archivo, al protegerlo se guardan los cambios en el repositorio. El icono del archivo indica si está desprotegido o protegido.

Las carpetas y los archivos aparecen con un icono diferente dependiendo de su estado:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Si selecciona el proyecto o una carpeta del proyecto, se seleccionan todos los archivos del objeto seleccionado. Para seleccionar varios objetos (archivos y carpetas) pulse la tecla **Ctrl** mientras hace clic en los objetos. La imagen siguiente muestra un proyecto que se desprotegió. Después se protegió el archivo `ExpReport.xml`.



Guardar cambios y rechazar cambios

Recuerde que al proteger un archivo puede elegir si se protegen los cambios o no. Es decir, puede guardar los cambios en el repositorio y seguir con el archivo desprotegido. Esto es muy útil si desea guardar cambios cada cierto tiempo y después seguir editando el archivo desprotegido.

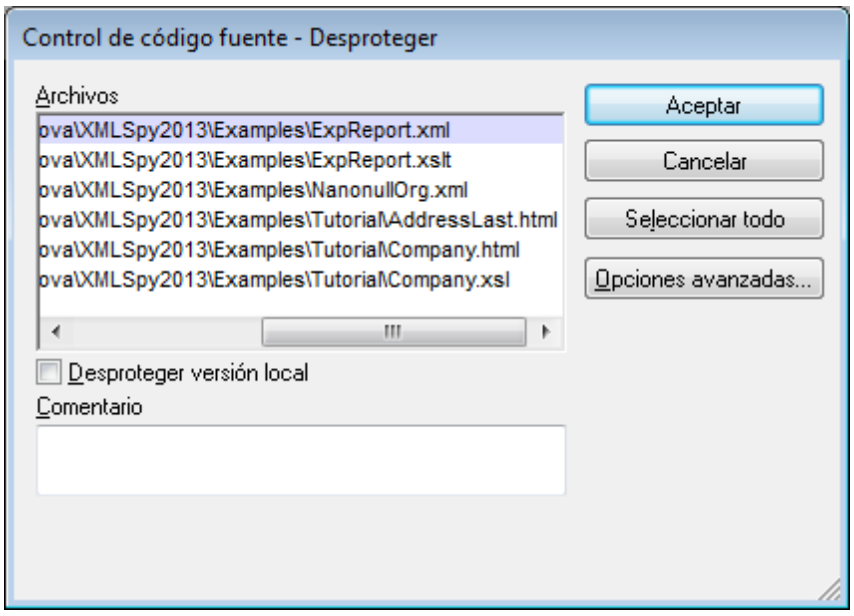
Si desprotegió un archivo y realizó cambios pero quiere rechazarlos por cualquier motivo, puede volver a la versión del documento guardada en el repositorio con solo hacer clic en el comando

© 2015 Altova GmbH

Proyecto | Control de código fuente | Deshacer desprotección.

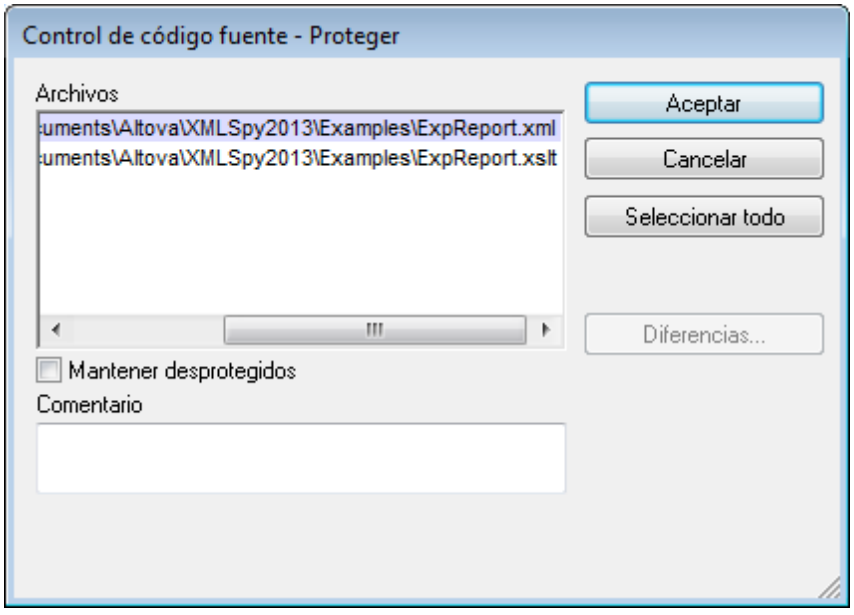
Desproteger

El cuadro de diálogo "Desproteger" (*imagen siguiente*) sirve para: (i) seleccionar los archivos que desea desproteger y (ii) seleccionar si se desprotege la versión del repositorio o la versión local.



Proteger

El cuadro de diálogo "Proteger" (*imagen siguiente*) sirve para: (i) seleccionar los archivos que desea proteger y (ii) mantener el archivo desprotegido.



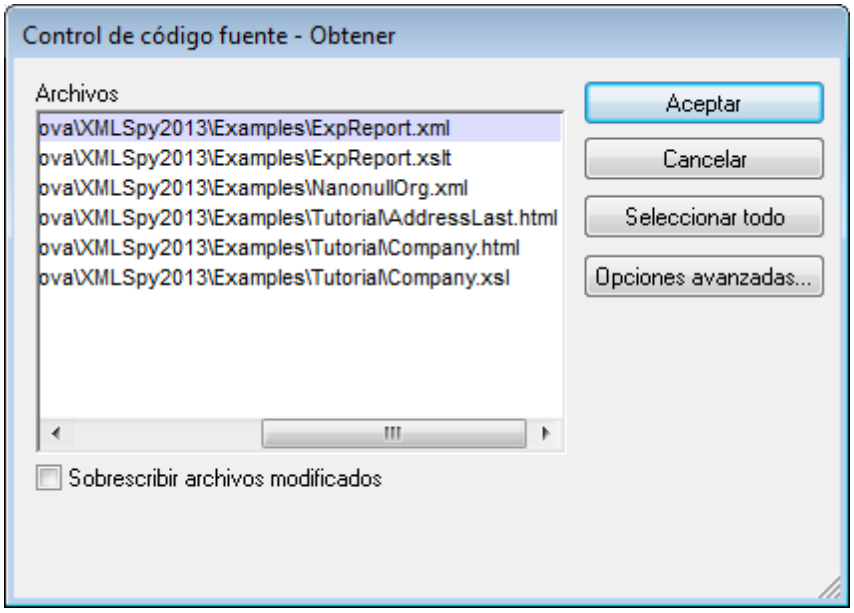
Nota: en estos dos cuadros de diálogo aparecen varios archivos si el objeto seleccionado (el proyecto o las carpetas de proyecto) contienen varios archivos.

© 2015 Altova GmbH

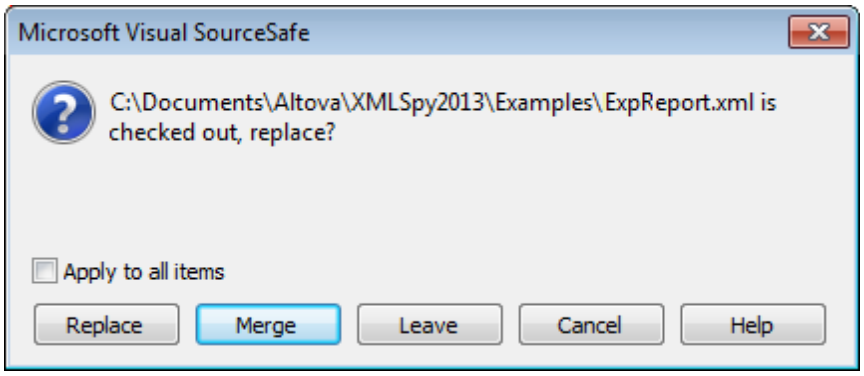
7.6.3 Obtener archivos como archivos de solo lectura

El comando **Obtener** (del menú **Proyecto | Control de código fuente**) recupera los archivos del repositorio como archivos de solo lectura. (Para poder editar un archivo, antes es necesario desprotegerlo.) El cuadro de diálogo "Obtener" enumera los archivos del objeto (proyecto o carpeta) sobre el que se ejecutó el comando **Obtener** (*imagen siguiente*). En el cuadro de diálogo puede marcar las casillas de los archivos que desea recuperar.

Nota: el comando **Obtener carpetas** sirve para seleccionar subcarpetas del repositorio (si su sistema de control de código fuente permite esta opción).

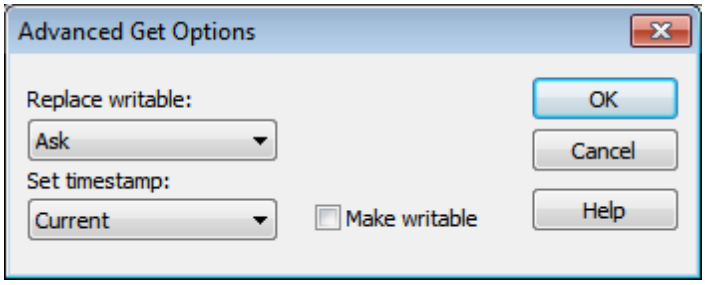


Si quiere, puede sobrescribir los archivos desprotegidos modificados marcando la opción *Sobrescribir archivos modificados* situada en la parte inferior del cuadro de diálogo. Al hacer clic en **Aceptar** los archivos se sobrescriben. Si alguno de los archivos sobrescritos está abierto, aparece un aviso preguntando si desea volver a cargar los archivos (botón **Volver a cargar**), cerrar los archivos (botón **Cerrar**) o conservar la vista actual del archivo (botón **Cancelar**).



Opciones avanzadas

El cuadro de diálogo "Opciones avanzadas" (*imagen siguiente*) se abre con el botón **Opciones avanzadas** del cuadro de diálogo "Obtener" (*primera imagen de este apartado*).



Aquí puede seleccionar (i) si reemplaza los archivos que se pueden escribir y que están desprotegidos, (ii) la marca de tiempo y (iii) si la propiedad de solo lectura del archivo recuperado se cambia para que el archivo se pueda escribir.

Obtener la versión más reciente

El comando **Obtener la versión más reciente** (del menú **Proyecto | Control de código fuente**) recupera y coloca en el directorio de trabajo la versión más reciente de los archivos seleccionados del control de código fuente. Los archivos se recuperan como archivos de solo lectura y no están desprotegidos. Este comando es igual que el comando **Obtener**, la única diferencia es que no aparece el cuadro de diálogo "Obtener".

Si los archivos seleccionados están desprotegidos, la acción que se toma dependerá de lo que haga su sistema de control en situaciones así. Por lo general, el sistema de control de código fuente le preguntará si desea reemplazar el archivo desprotegido, combinarlo o dejarlo tal y como está.

Nota: cuando seleccione este comando para una carpeta, el comando se aplica de forma recursiva. Esto significa que se aplica a todos los archivos situados dentro de la carpeta actual.

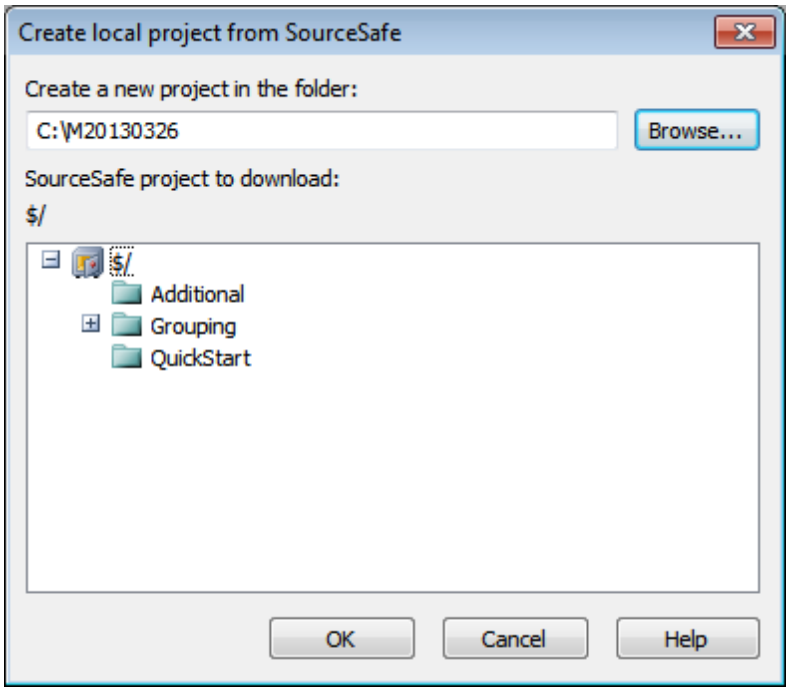
© 2015 Altova GmbH

7.6.4 Copiar y compartir desde el control de código fuente

El comando **Abrir desde el control de código fuente** crea un proyecto de aplicación nuevo a partir de un proyecto bajo control de código fuente.

Estos son los pasos para crear el proyecto:

1. Dependiendo del sistema de control de código fuente que utilice, quizás sea necesario comprobar que ningún archivo del proyecto está desprotegido antes de crear un proyecto nuevo.
2. No es necesario tener abierto un proyecto en la aplicación.
3. Seleccione el comando **Proyecto | Control de código fuente | Abrir desde el control de código fuente**.
4. El sistema de control de código fuente que tiene instalado muestra sus cuadros de diálogo de verificación y conexión. Conéctese al repositorio pertinente (es decir, a la carpeta enlazada del repositorio que equivale a la carpeta local).
5. En el cuadro de diálogo que aparece (*imagen siguiente*) busque la carpeta local en la que desea copiar el contenido de la carpeta del repositorio (a la que se acaba de conectar). En la imagen siguiente, por ejemplo, la carpeta asociada del repositorio se llama `MiProyecto` y se representa con el signo `$`. La carpeta local es `C:\M20130326`.



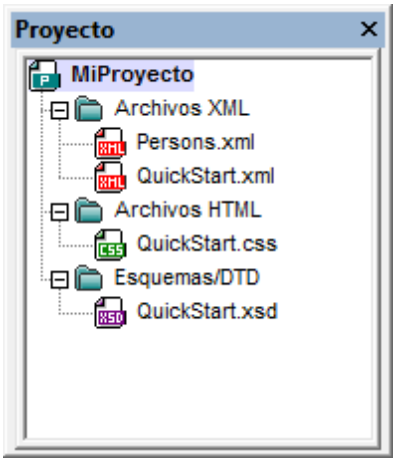
6. Haga clic en **OK**. El contenido de la carpeta asociada (`MiProyecto`) se copiará a la carpeta local `C:\M20130326`. y aparece un cuadro de diálogo pidiendo que seleccione el archivo de proyecto (archivo `.spp`) que desea crear como proyecto nuevo.
7. Seleccione el archivo `.spp` que se copió a la carpeta local. En nuestro ejemplo, se trata del archivo `MiProyecto.spp` situado en la carpeta `C:\M20130326`. En la aplicación se crea un proyecto nuevo llamado `MiProyecto` y este proyecto aparece en la ventana Proyecto. Los archivos del proyecto están en la carpeta `C:\M20130326`.

Compartir desde el control de código fuente

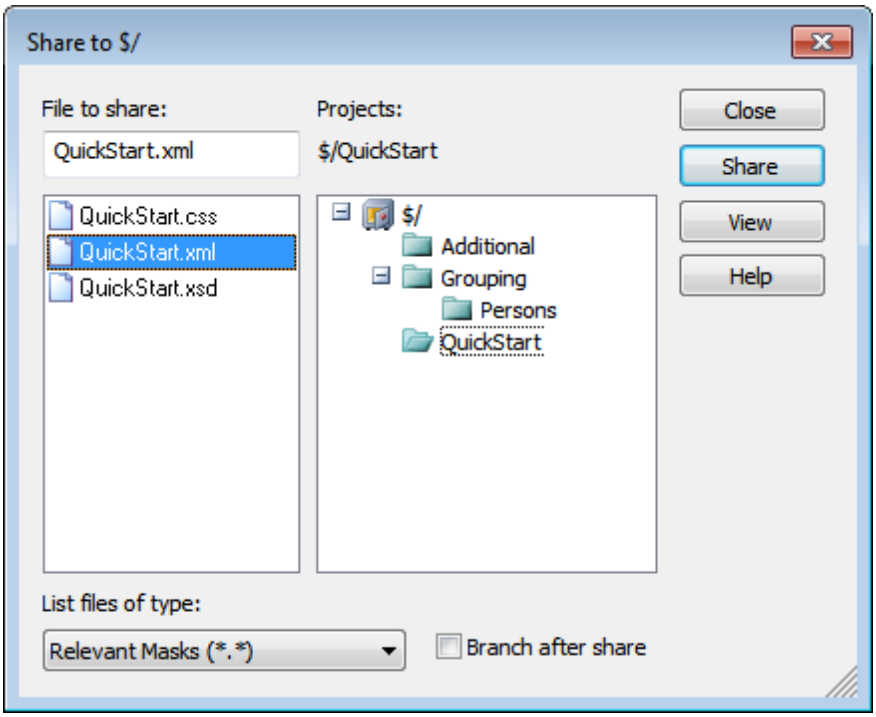
El comando **Compartir desde el control de código fuente** está disponible si su sistema de

control de código fuente permite compartir contenido. Por ejemplo, puede compartir un archivo para que esté disponible en varias ubicaciones locales. Si se realizan cambios en uno de estos archivos locales, los cambios se reflejan en todas las versiones "compartidas" del archivo.

En la ventana Proyecto seleccione el proyecto (*imagen siguiente*) y después haga clic en **Compartir desde el control de código fuente**.



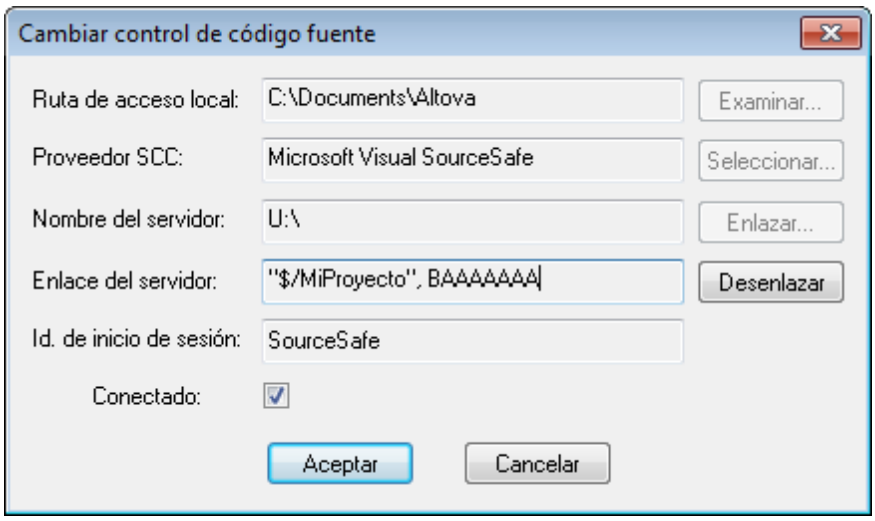
Aparece el cuadro de diálogo "Share To [carpeta]" (*imagen siguiente*).



Ahora puede seleccionar los archivos que desea compartir. Primero seleccione en el panel derecho la carpeta donde están los archivos. Los archivos de la carpeta elegida aparecen en el panel izquierdo. Seleccione el archivo que desea compartir (para seleccionar varios pulse la tecla **Ctrl** mientras hace clic en los archivos). Los archivos seleccionados aparecen en el cuadro situado en la parte superior izquierda del diálogo. Los archivos desaparecen del panel izquierdo. Haga clic en **Share** (*compartir*) y después en **Close** (cerrar) para copiar los archivos seleccionados a la carpeta local compartida.

© 2015 Altova GmbH

El nombre de la carpeta compartida aparece en el nombre del cuadro de diálogo. En la imagen anterior, por ejemplo, se trata de la carpeta local (porque el signo $ representa la carpeta del repositorio a la que está enlazada la carpeta local). En el cuadro de diálogo "Cambiar control de código fuente" puede ver y configurar la carpeta compartida (*imagen siguiente*) cambiando la ruta de acceso local y el enlace de servidor.



Para más información consulte la documentación de su sistema de control de código fuente.

7.6.5 Cambiar control de código fuente

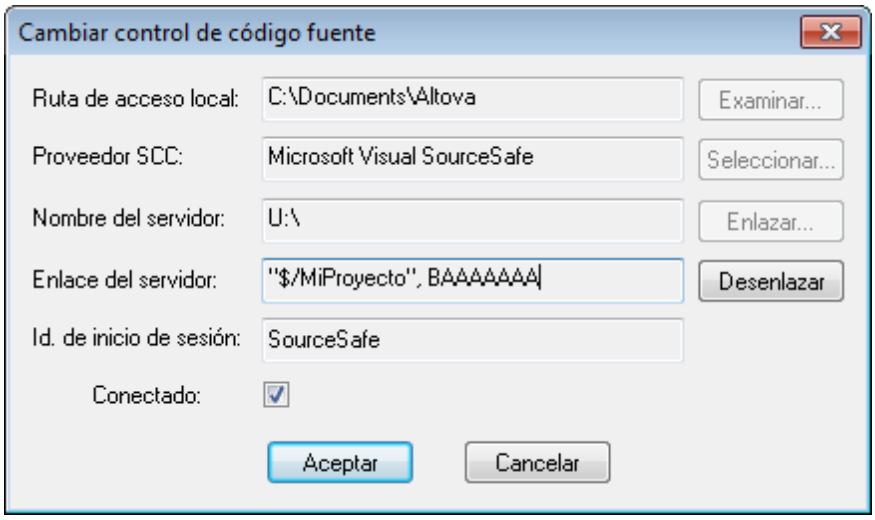
El control de código fuente se puede cambiar con dos comandos diferentes del menú **Proyecto | Control de código fuente**:

- Administrador del control de código fuente, que abre el programa de control de código fuente y permite configurar bases de datos y enlaces.
- Cambiar control de código fuente, que abre el cuadro de diálogo del mismo nombre, donde puede cambiar de sistema (ver la descripción que aparece más abajo).

El enlace actual es lo que usa el proyecto de aplicación activo para conectarse a la BD del control de código fuente. Por tanto, en enlace actual debe ser correcto. Esto significa que el archivo de proyecto de la aplicación (archivo `.spp`) debe estar en la carpeta local y la carpeta enlazada del repositorio debe ser la base de datos donde se almacenan los archivos de este proyecto. Por lo general, la carpeta enlazada y sus subcarpetas se corresponden con la carpeta de la copia de trabajo local y sus subcarpetas.

En el cuadro de diálogo "Cambiar control de código fuente" (*imagen siguiente*) puede cambiar el sistema de control (*Proveedor SCC*), la carpeta local (*Ruta de acceso local*) y el enlace al repositorio (*Nombre del servidor* y *Enlace del servidor*).

Las opciones solamente se pueden cambiar después de desenlazar el enlace actual. Para ello, pulse el botón **Desenlazar**. Ahora puede editar las demás opciones.



1. Con el botón **Examinar** puede buscar la carpeta local y con el botón Seleccionar puede seleccionar uno de los sistemas de control de código fuente instalados.
2. Después puede enlazar la carpeta local a una BD del repositorio. Para ello pulse el botón **Enlazar**. Aparece el cuadro de diálogo de conexión de sus sistema de control.
3. Si escribió un *Id. de inicio de sesión*, este identificador se pasa al sistema de control. De lo contrario, deberá escribir sus datos de inicio de sesión en el diálogo de conexión.
4. Seleccione la BD del repositorio a la que desea enlazar esta carpeta local.
5. Cuando termine haga clic en **Aceptar**.

© 2015 Altova GmbH

8 Authentic Desktop en Visual Studio

Authentic Desktop se puede integrar en las versiones 2005/2008/2010/2012/2013/2015 del entorno IDE Microsoft Visual Studio. Esto hace posible incorporar las avanzadas funciones de edición XML al avanzado entorno de desarrollo de Visual Studio.

En esta sección describimos:

- El proceso general de instalación y la integración del complemento de Authentic Desktop para Visual Studio.
- Las diferencias entre Authentic Desktop y el complemento de Authentic Desktop para Visual Studio.

8.1 Instalar el complemento de Authentic Desktop para Visual Studio

Siga estas instrucciones para instalar el complemento de Authentic Desktop para Visual Studio:

1. Instale Microsoft Visual Studio 2005/2008/2010/2012/2013/2015.
2. Instale Authentic Desktop (Enterprise or Professional Edition).
3. Descargue y ejecute el paquete de integración de Authentic Desktop. Este paquete está disponible en la página de descargas de Authentic Desktop (ediciones Enterprise y Professional) de www.altova.com. **Nota importante:** debe usar el paquete de integración correspondiente a su versión de Authentic Desktop (la versión actual es 2016).

Una vez instalado el paquete de integración puede empezar a usar Authentic Desktop en el entorno de Visual Studio.

Cómo se habilita el complemento

Si el complemento no se habilita automáticamente durante el proceso de instalación, entonces siga estas instrucciones:

1. Navegue hasta el directorio donde se instaló el ejecutable de Visual Studio, por ejemplo en `C:\Archivos de programa\MS Visual Studio\Common7\IDE`
2. Escriba este comando en la línea de comandos `devenv.exe /setup`.
3. Espere a que termine el proceso antes de usar la aplicación dentro de Visual Studio.

Nota: las imágenes y capturas de pantalla de esta sección son de Visual Studio 2010. Si utiliza otra versión de Visual Studio, puede que encuentre diferencias entre estas imágenes y su versión.

© 2015 Altova GmbH

8.2 Diferencias entre la versión independiente y la versión para Visual Studio

Este apartado enumera las diferencias entre el complemento de Authentic Desktop para Visual Studio y la versión independiente de Authentic Desktop.

Ayudantes de entrada (Ventanas de herramientas en Visual Studio)
Los ayudantes de entrada de Authentic Desktop se llaman *Ventanas de herramientas* en Visual Studio. Es necesario tener en cuenta estos aspectos:

- Puede arrastrar las ventanas de los ayudantes de entrada y colocarlas en cualquier posición del entorno de desarrollo.
- Puede seguir personalizando la interfaz si hace clic con el botón secundario en una pestaña de un ayudante de entrada. El menú contextual que aparece incluye opciones para acoplar la ventana, ocultarla, ocultarla automáticamente o hacerla flotante.

> **Nota:** para más información sobre los ayudantes de entrada de Authentic Desktop consulte la sección Introducción.

Comandos de Authentic Desktop como comandos de Visual Studio
Algunos comandos de Authentic Desktop son comandos de Visual Studio en la interfaz de Visual Studio. Se trata de estos comandos:

- ***Deshacer/Rehacer:*** estos comandos de Visual Studio afectan a todas las acciones realizadas en el entorno de desarrollo Visual Studio.
- ***Proyectos:*** los proyectos de Authentic Desktop se consideran proyectos de Visual Studio.
- ***Personalizar las barras de herramientas/Personalizar los comandos:*** las pestañas *Barras de herramientas* y *Comandos* del cuadro de diálogo Personalizar (**Herramientas | Personalizar**) incluye tanto los comandos de Visual Studio como los de Authentic Desktop.
- ***Vistas:*** en el menú **Ver** el comando **Authentic Desktop** incluye opciones para activar/desactivar las ventanas de los ayudantes de entrada y otras barras laterales, para cambiar de vista de edición y para activar/desactivar determinadas guías de edición.
- ***Ayuda de Authentic Desktop:*** este menú de Authentic Desktop es un submenú del menú **Ayuda** de Visual Studio.

Nota: para editar un archivo XML con el complemento de Authentic para Visual Studio, seleccione el comando **Archivo | Abrir**. Después, en el cuadro de diálogo "Abrir archivo", utilice la opción **Abrir con** para seleccionar el complemento de Authentic.

9 Authentic Desktop en Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto que incorpora varios tipos de aplicaciones en forma de complementos.

Gracias al complemento de Authentic Desktop para Eclipse puede usar todas las funciones de Authentic Desktop desde la plataforma Eclipse 4.2 / 4.3 / 4.4. El complemento para Eclipse está disponible para todas las plataformas Windows. En esta sección explicamos cómo instalar el complemento y cómo configurar la perspectiva Authentic. Después de hacer esto los componentes de la interfaz de Authentic Desktop y los comandos de menú de Authentic Desktop estarán disponibles en la interfaz de Eclipse.

© 2015 Altova GmbH

9.1 Instalar el complemento de Authentic Desktop para Eclipse

Antes de instalar el complemento de Authentic Desktop para Eclipse, asegúrese de que estas aplicaciones están instaladas:

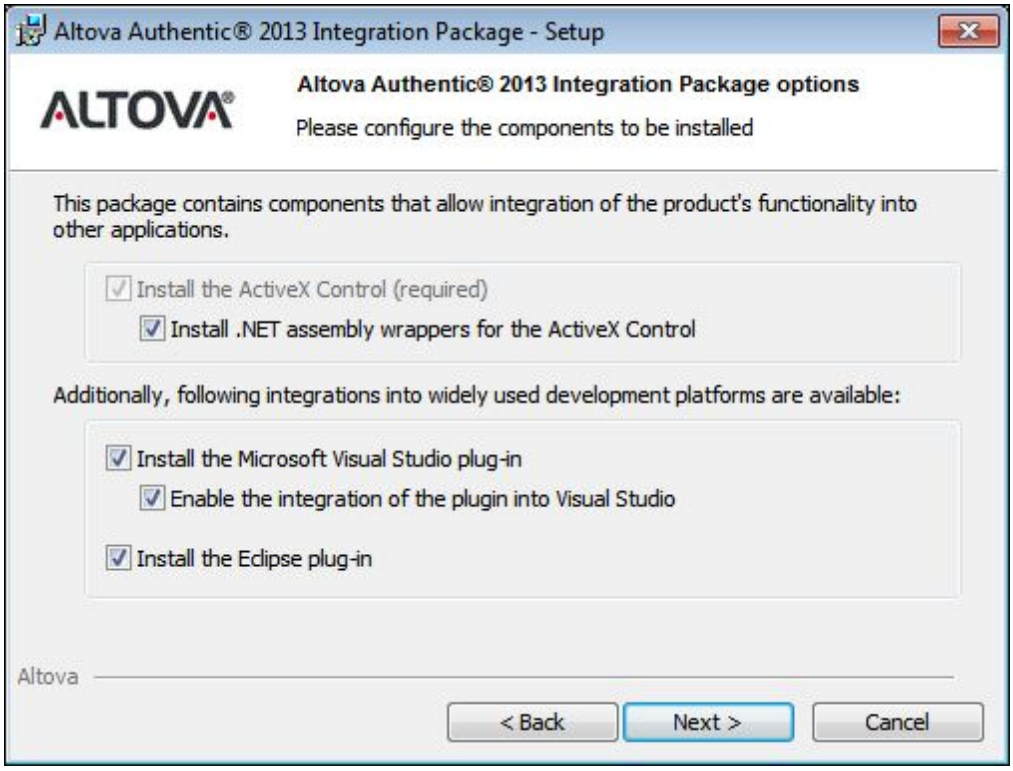
- Authentic Desktop Enterprise o Professional Edition.
- Java SE Runtime Environment 6.0 (JRE 6.0) o superior, que es necesario para que funcione Eclipse. Consulte el sitio web de Eclipse para obtener más información. Instale la versión de 32 o 64 bits dependiendo de la versión de Authentic Desktop que tenga instalada.
- Eclipse Platform 4.2 / 4.3 / 4.4. Instale la versión de 32 o 64 bits dependiendo de la versión de Authentic Desktop que tenga instalada.

Después de instalar todas estas aplicaciones puede instalar el complemento de Authentic Desktop para Eclipse que incluye el paquete de integración de Authentic Desktop (*ver más abajo*).

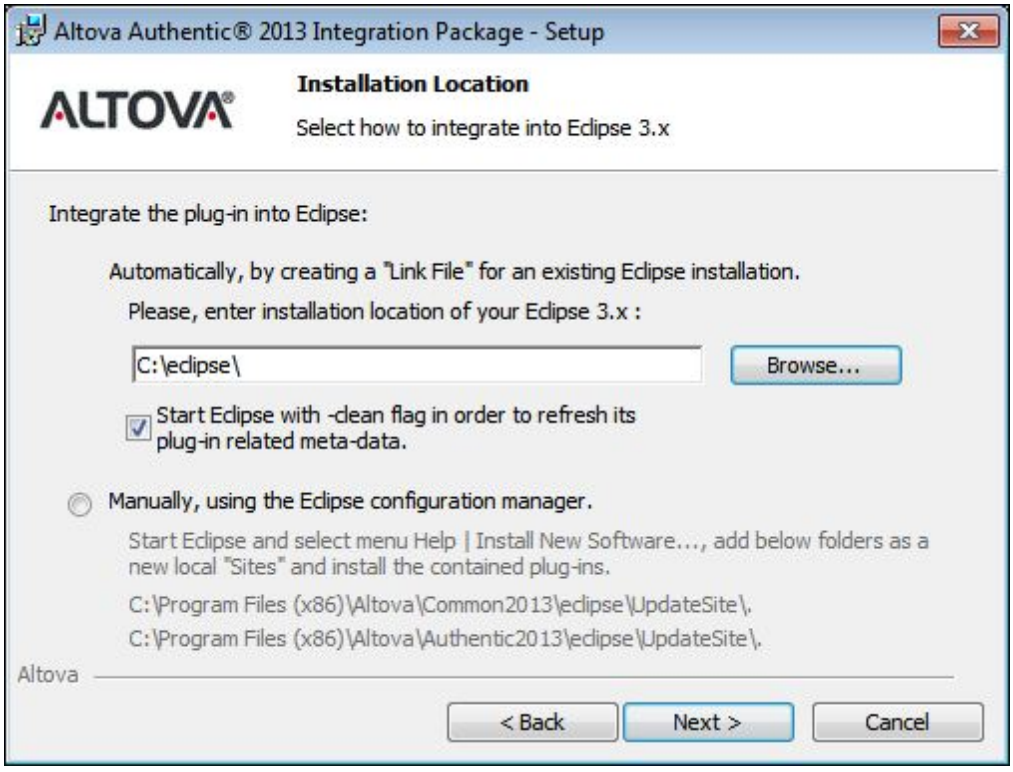
Paquete de integración de Authentic Desktop

El complemento de Authentic Desktop para Eclipse está en el paquete de integración de Authentic Desktop y se instala durante la instalación del paquete de integración de Authentic Desktop. Estas son las instrucciones de instalación:

1. Compruebe que tiene instalado Authentic Desktop (Enterprise o Professional Edition), JRE y Eclipse (*ver más arriba*).
2. En la página de descargas de componentes del sitio web de Altova descargue e instale el paquete de integración de Authentic Desktop. Durante la instalación hay dos pasos muy importantes que se describen en los puntos 3 y 4.
3. Durante la instalación del paquete de integración de Authentic Desktop aparece una pantalla que le pregunta si desea instalar el complemento de Authentic Desktop para Eclipse (*imagen siguiente*). Marque la casilla y haga clic en **Siguiente**.

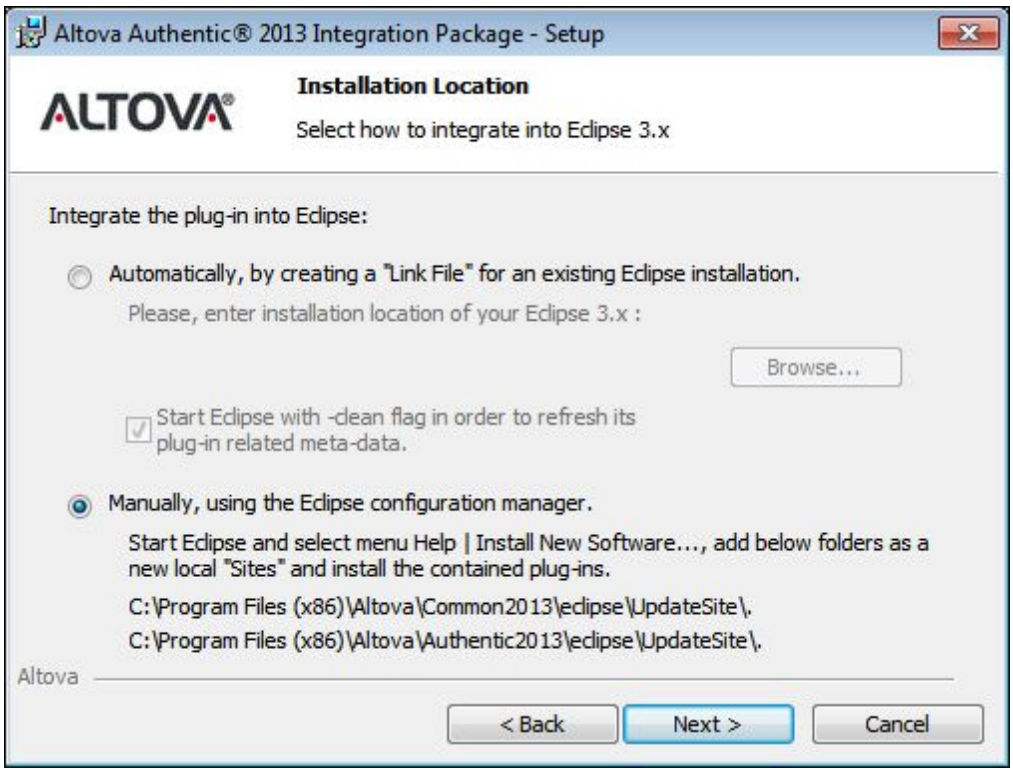

Altova Authentic® 2013 Integration Package - Setup
ALTOVA
Altova Authentic® 2013 Integration Package options
Please configure the components to be installed
This package contains components that allow integration of the product's functionality into other applications.
Install the ActiveX Control (required)
Install .NET assembly wrappers for the ActiveX Control
Additionally, following integrations into widely used development platforms are available:
Install the Microsoft Visual Studio plug-in
Enable the integration of the plugin into Visual Studio
Install the Eclipse plug-in
Altova
< Back
Next >
Cancel

4. En la siguiente pantalla (*imagen siguiente*) puede indicar si el asistente para la instalación debe integrar el complemento de Authentic Desktop para Eclipse durante la instalación (opción *Automáticamente*) o si prefiere integrarlo de forma manual más adelante.


Altova Authentic® 2013 Integration Package - Setup
ALTOVA
Installation Location
Select how to integrate into Eclipse 3.x
Integrate the plug-in into Eclipse:
Automatically, by creating a "Link File" for an existing Eclipse installation.
Please, enter installation location of your Eclipse 3.x :
C:\eclipse\
Browse...
Start Eclipse with -clean flag in order to refresh its plug-in related meta-data.
Manually, using the Eclipse configuration manager.
Start Eclipse and select menu Help | Install New Software..., add below folders as a new local "Sites" and install the contained plug-ins.
C:\Program Files (x86)\Altova\Common2013\eclipse\UpdateSite\.
C:\Program Files (x86)\Altova\Authentic2013\eclipse\UpdateSite\.
Altova
< Back
Next >
Cancel

 © 2015 Altova GmbH

Recomendamos que elija la opción *Automáticamente* para que el asistente para la instalación se encargue de integrar el complemento. Seleccione la opción *Automáticamente* y después busque la carpeta donde se encuentra el ejecutable de Eclipse (`eclipse.exe`). Después haga clic en **Siguiente**. Si prefiere integrar el complemento de Authentic Desktop para Eclipse de forma manual, seleccione la opción *De forma manual* (*imagen siguiente*). Más abajo explicamos cómo se integra el complemento de forma manual.

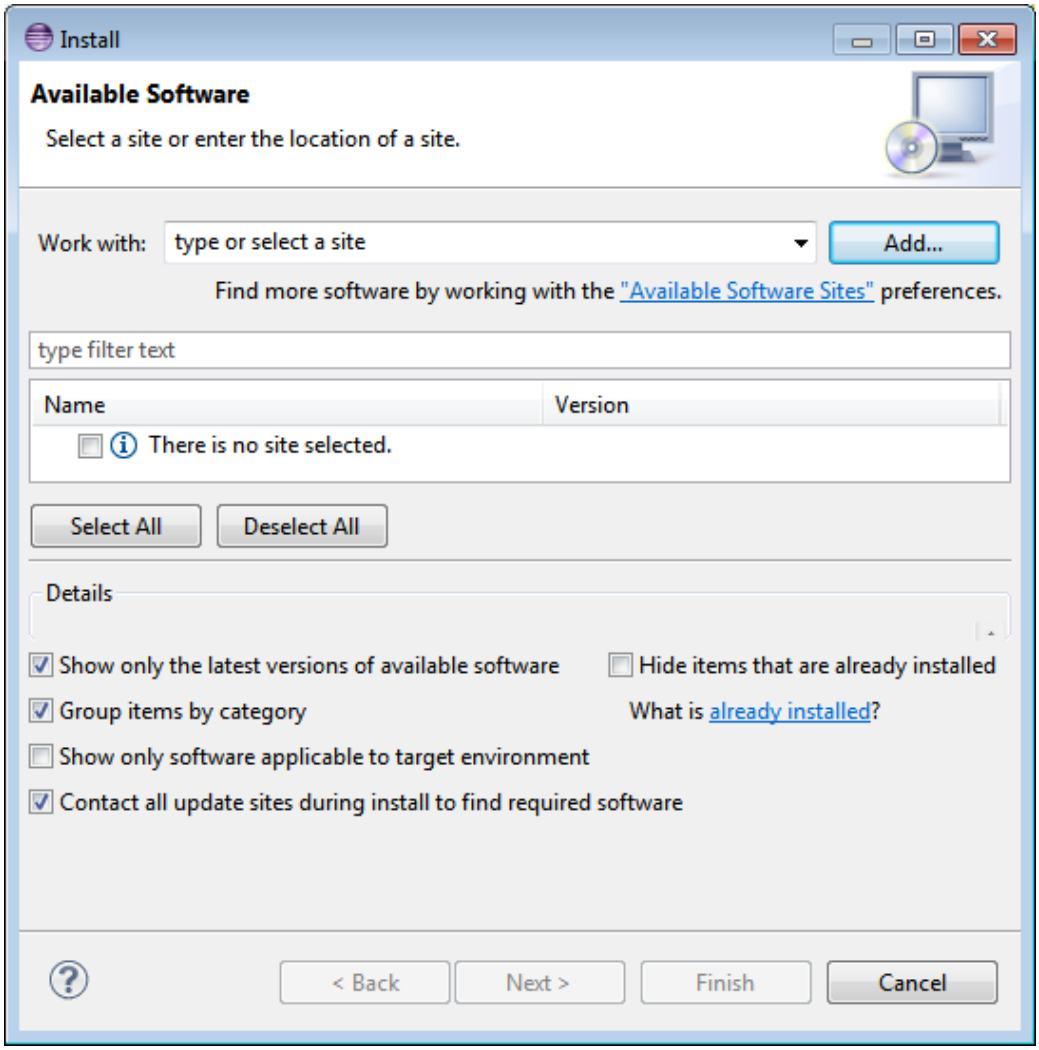

Altova Authentic® 2013 Integration Package - Setup
ALTOVA
Installation Location
Select how to integrate into Eclipse 3.x
Integrate the plug-in into Eclipse:
Automatically, by creating a "Link File" for an existing Eclipse installation.
Please, enter installation location of your Eclipse 3.x :
Browse...
Start Eclipse with -clean flag in order to refresh its plug-in related meta-data.
Manually, using the Eclipse configuration manager.
Start Eclipse and select menu Help | Install New Software..., add below folders as a new local "Sites" and install the contained plug-ins.
C:\Program Files (x86)\Altova\Common2013\eclipse\UpdateSite\.
C:\Program Files (x86)\Altova\Authentic2013\eclipse\UpdateSite\.
Altova
< Back
Next >
Cancel

5. Finalice la instalación. Si seleccionó la integración automática, el complemento de Authentic Desktop para Eclipse se integrará en Eclipse y estará disponible la próxima vez que inicie Eclipse.

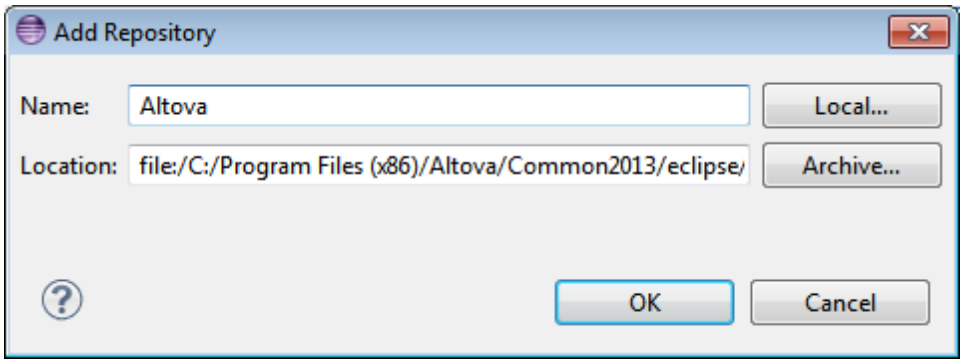
Integrar el complemento de Authentic Desktop en Eclipse de forma manual

Para integrar el complemento de Authentic Desktop en Eclipse de forma manual:

1. En Eclipse haga clic en el comando **Help | Install New Software** (Ayuda | Instalar software nuevo).
2. En el cuadro de diálogo que aparece (*imagen siguiente*) haga clic en el botón **Add** (Agregar).


Install
Available Software
Select a site or enter the location of a site.
Work with:
type or select a site
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
There is no site selected.
Select All
Deselect All
Details
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Show only software applicable to target environment
Contact all update sites during install to find required software
< Back
Next >
Finish
Cancel

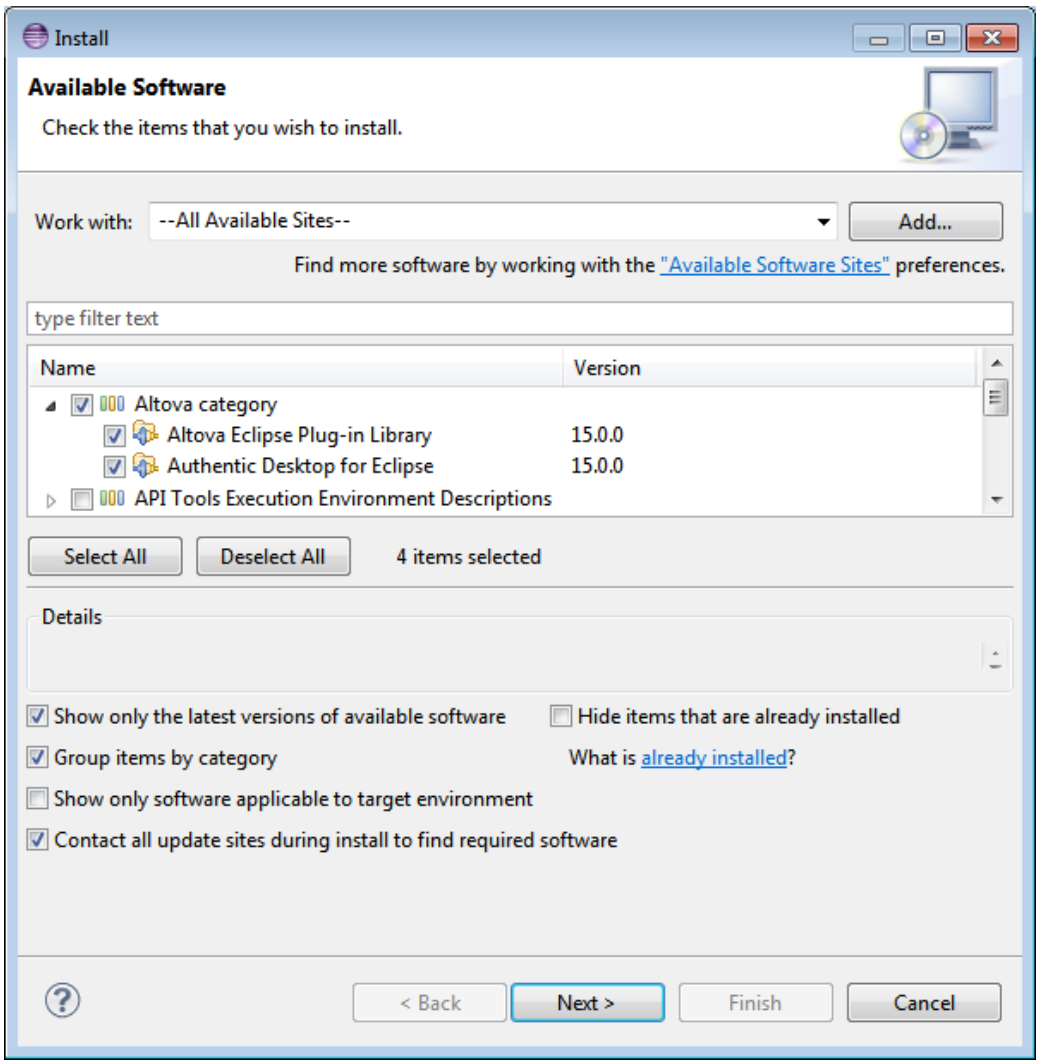
3. En el cuadro de diálogo que aparece, llamado "Add Repository" (Agregar repositorio, *imagen siguiente*) haga clic en el botón **Local**.
4. Busque la carpeta `c:\Archivos de programa\Altova\Common2016\eclipse\UpdateSite` y selecciónela. Escriba un nombre para el sitio (p. ej. Altova) y haga clic en **OK**.



5. Repita los pasos 2, 3 y 4 pero esta vez seleccione la carpeta `c:\Archivos de programa\Altova\XMLSpy2016\eclipse\UpdateSite` y escriba un nombre como "Altova Authentic Desktop".

© 2015 Altova GmbH

6. En el cuadro combinado *Work With* (Trabajar con) del cuadro de diálogo "Install" seleccione la opción *All Available Sites* (Todos los sitios disponibles, *imagen siguiente*). Como resultado aparecen todos los complementos disponibles en el panel inferior. Marque la casilla de nivel superior de la carpeta *Altova category* (*imagen siguiente*). Después haga clic en **Next** (Siguiente).



7. Aparece una pantalla llamada "Install Details" (detalles de la instalación) en la que puede repasar los elementos que se van a instalar. Haga clic en **Next** (Siguiente) para continuar.
8. Aparece la pantalla "Review Licenses" (Repasar licencias). Seleccione la opción *I accept the terms of the license agreement* (Acepto el contrato de licencia). Para el complemento de Authentic Desktop no es necesario otro contrato de licencia (basta con la clave de licencia de Authentic Desktop Enterprise o Professional Edition). Para terminar haga clic en el botón **Finish** (Finalizar).

Si tiene problemas con el complemento (por ejemplo, si faltan iconos), inicie Eclipse desde la línea de comandos con la marca `-clean`.

Versión instalada actualmente

Para comprobar qué versión tiene instalada del complemento de Authentic Desktop para Eclipse, en Eclipse seleccione la opción de menú **Help | About Eclipse**. Después seleccione el icono de Authentic Desktop.

© 2015 Altova GmbH

9.2 Puntos de entrada de Authentic Desktop en Eclipse

Para acceder a las funciones de Authentic Desktop puede usar estos puntos de entrada en Eclipse:

- Perspectiva Authentic Desktop, que ofrece las funciones de la interfaz de Authentic Desktop dentro de la interfaz de Eclipse.
- Botones de la barra de herramientas Authentic Desktop

Perspectiva de Authentic Desktop

En Eclipse una perspectiva es una vista de la interfaz del usuario configurada con función de operador. Cuando el complemento de Authentic Desktop está integrado en Eclipse, se crea automáticamente una perspectiva Authentic Desktop predeterminada. Esta perspectiva es una interfaz gráfica del usuario que incluye los elementos de la interfaz de Authentic Desktop: sus vistas de edición, menús, ayudantes de entrada y otras barras laterales.

Al abrir un tipo de archivo asociado a Authentic Desktop (por ejemplo, `.xml`), el archivo se puede editar en la perspectiva Authentic Desktop. También puede abrir archivos de otro tipo en otra perspectiva de Eclipse. Además, puede cambiar de perspectiva en todo momento, lo cual permite editar o procesar el archivo en otro entorno. Por tanto, las perspectivas tienen dos ventajas:

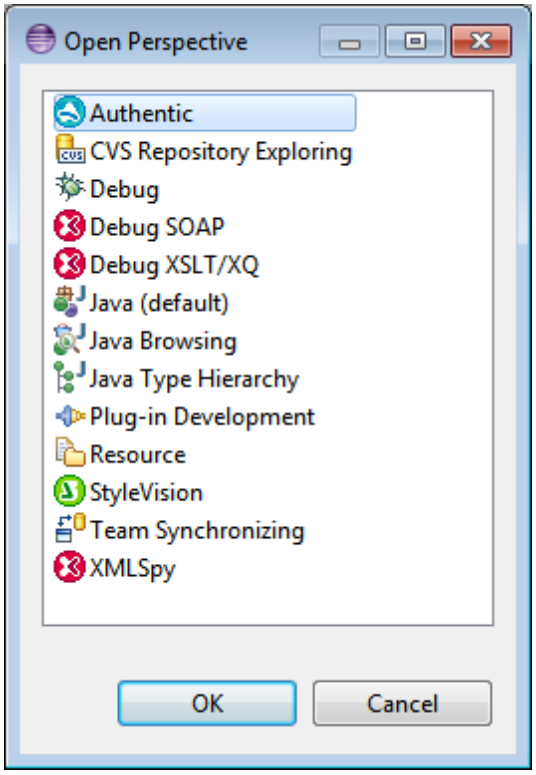
1. Permiten cambiar el entorno de trabajo para el archivo activo
2. Y permiten cambiar de un archivo a otro sin necesidad de abrir un entorno de desarrollo nuevo (el entorno asociado está disponible en la perspectiva)

Para trabajar con la perspectiva Authentic Desktop es necesario:

- Cambiar a la perspectiva Authentic Desktop.
- Configurar las preferencias de la perspectiva Authentic Desktop.
- Personalizar la perspectiva Authentic Desktop.

Cambiar a la perspectiva Authentic Desktop

En Eclipse seleccione el comando **Window | Open Perspective | Other**. En el diálogo que aparece (*imagen siguiente*) seleccione **Authentic Desktop** y haga clic en **OK**.



La ventana vacía o el documento activo tendrán ahora la perspectiva Authentic Desktop. Así es como se cambia de perspectiva desde el menú. Para cambiar más rápido de una perspectiva a otra se puede incluir la perspectiva deseada en el submenú **Open Perspective**, sobre la opción **Other**. Esto se hace en el cuadro de diálogo de personalización.

También puede cambiar de perspectiva al abrir un archivo o activar un archivo. La perspectiva asociada al tipo de archivo se abre automáticamente al abrir el archivo por primera vez. Antes de que cambie la perspectiva, aparece un cuadro de diálogo donde debe indicar si desea abrir automáticamente la perspectiva asociada a este tipo de archivo..

Marque la casilla *Do Not Ask Again* (No volver a preguntar) si desea asociar la perspectiva a ese tipo de archivo sin tener que confirmarlo cada vez que abra un archivo de este tipo. Después haga clic en **OK**.

Configurar las preferencias de la perspectiva Authentic Desktop

Las preferencias de una perspectiva incluyen: (i) una opción para cambiar automáticamente de perspectiva al abrir un archivo de un tipo determinado (ver párrafo anterior) y (ii) opciones para incluir o excluir algunas barras de herramientas de Authentic Desktop.

En la lista de perspectivas del panel de la izquierda, seleccione Authentic Desktop y después las opciones que desea usar. Para terminar haga clic en **OK**.

Personalizar la perspectiva Authentic Desktop

En Eclipse puede configurar qué accesos directos y comandos aparecen en cada perspectiva. Para abrir el cuadro de diálogo para personalizar la perspectiva (*imagen siguiente*), active la perspectiva que desea personalizar (en este caso la perspectiva Authentic Desktop) y seleccione el comando **Window | Customize Perspective**.

© 2015 Altova GmbH

En la pestaña *Commands* puede agregar grupos de comandos a menús primarios y a la barra de herramientas. Si desea habilitar un grupo de comandos, marque la casilla del grupo.

En la pestaña *Shortcuts* puede definir accesos directos para los submenús. Seleccione el submenú en el cuadro combinado *Submenus*, después seleccione una categoría de accesos directos y por último seleccione los accesos directos que desea incluir en la perspectiva.

Haga clic en **OK** para terminar de personalizar la perspectiva.

Menú y barra de herramientas de Authentic Desktop

El menú de **Authentic Desktop** contiene comandos que resultan relevantes incluso cuando el tipo de documento que reconoce Authentic Desktop no se puede abrir en Eclipse. Algunos de estos comandos aparecen en el menú **Archivo** de la versión independiente **de** Authentic Desktop.

La barra de herramientas de Authentic Desktop contiene los siguientes botones (*consulte la siguiente captura de pantalla*)

Estos comandos sirven (de izquierda a derecha) para (i) abrir la Ayuda de Authentic Desktop y (ii) acceder a los comandos de Authentic Desktop (como alternativa de acceso a los mismos desde el menú de **Authentic Desktop**.

10 Comandos de menú

La sección ***Comandos de menú*** ofrece una descripción completa de todos los comandos y menús de Authentic Desktop y explica cómo se utilizan. Si tiene alguna duda y no encuentra lo que busca en la *Comandos de menú* ni en el resto de la documentación, consulte la sección de preguntas frecuentes y foros de debate del sitio web de Altova. Si sigue sin encontrar lo que busca, póngase en contacto con el centro de soporte técnico de Altova.

Los comandos estándar de Windows (**Abrir**, **Guardar**, **Cortar**, **Copiar**, **Pegar**, etc.) están en los menús **Archivo** y **Edición**. Estos menús también incluyen comandos relacionados con XML e Internet.

© 2015 Altova GmbH

10.1 Menú Archivo

Los comandos del menú **Archivo** sirven para realizar operaciones con archivos. Además de los comandos estándar **Nuevo**, **Abrir**, **Guardar**, **Imprimir**, **Configurar impresión** y **Salir**, el menú **Archivo** incluye comandos característicos de Authentic Desktop.

10.1.1 Nuevo

En este apartado:

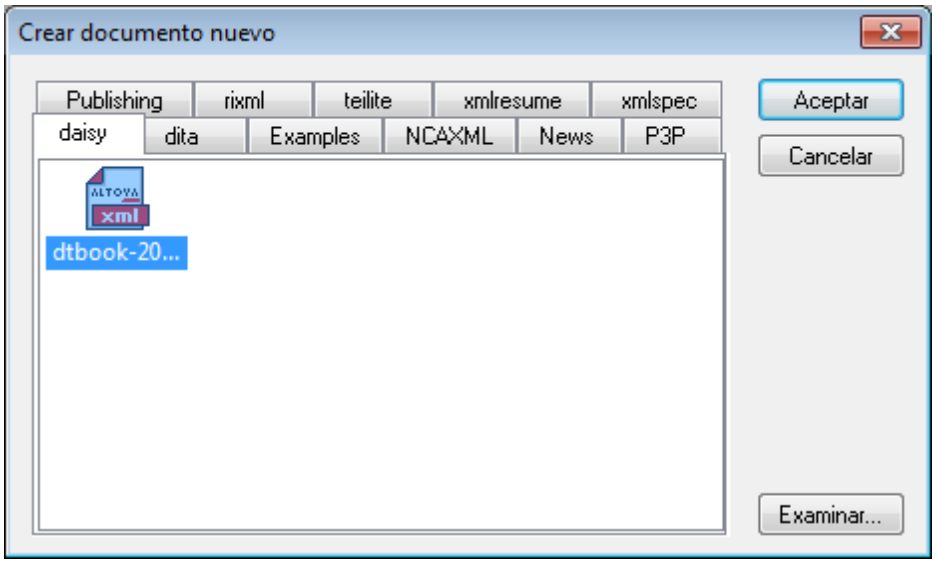
- Icono y acceso rápido
- Descripción

Icono y acceso rápido

Icono:	
Acceso rápido:	**Ctrl+N**

Descripción

Este comando sirve para abrir una plantilla de documento XML nuevo en la vista Authentic. La plantilla de documento XML está basada en un archivo SPS (StyleVision Power Stylesheet) y se abre seleccionando el archivo SPS en el cuadro de diálogo "Crear documento nuevo" (*imagen siguiente*). Tras seleccionar el archivo SPS haga clic en **Aceptar**. La plantilla de documento XML definida para el archivo SPS elegido se abre en la vista Authentic.



El cuadro de diálogo "Crear documento nuevo" ofrece varias plantillas de documento XML basadas en esquemas y archivos DTD frecuentes. Si lo prefiere, también puede buscar archivos SPS personalizados que tengan asignado un archivo XML. Los archivos SPS se crean con Altova StyleVision, una herramienta con la que puede diseñar plantillas de documento XML a partir de un esquema XML o un archivo DTD. Tras diseñar el archivo SPS en StyleVision puede asignarle un archivo XML como plantilla XML. Los datos de esta plantilla XML aportan los datos de partida de la plantilla de documento que se abre en la vista Authentic de Authentic Desktop.

© 2015 Altova GmbH

La nueva plantilla de documento XML las propiedades de presentación definidas en el archivo SPS y los datos del archivo XML que seleccionó como plantilla XML. El usuario de la vista Authentic puede editar la plantilla de documento XML en una interfaz gráfica WYSIWYG y guardar los cambios en un archivo XML.

10.1.2 Abrir

Icono y acceso rápido

Comando	*Icono*	*Acceso rápido*
Abrir		**Ctrl+O**

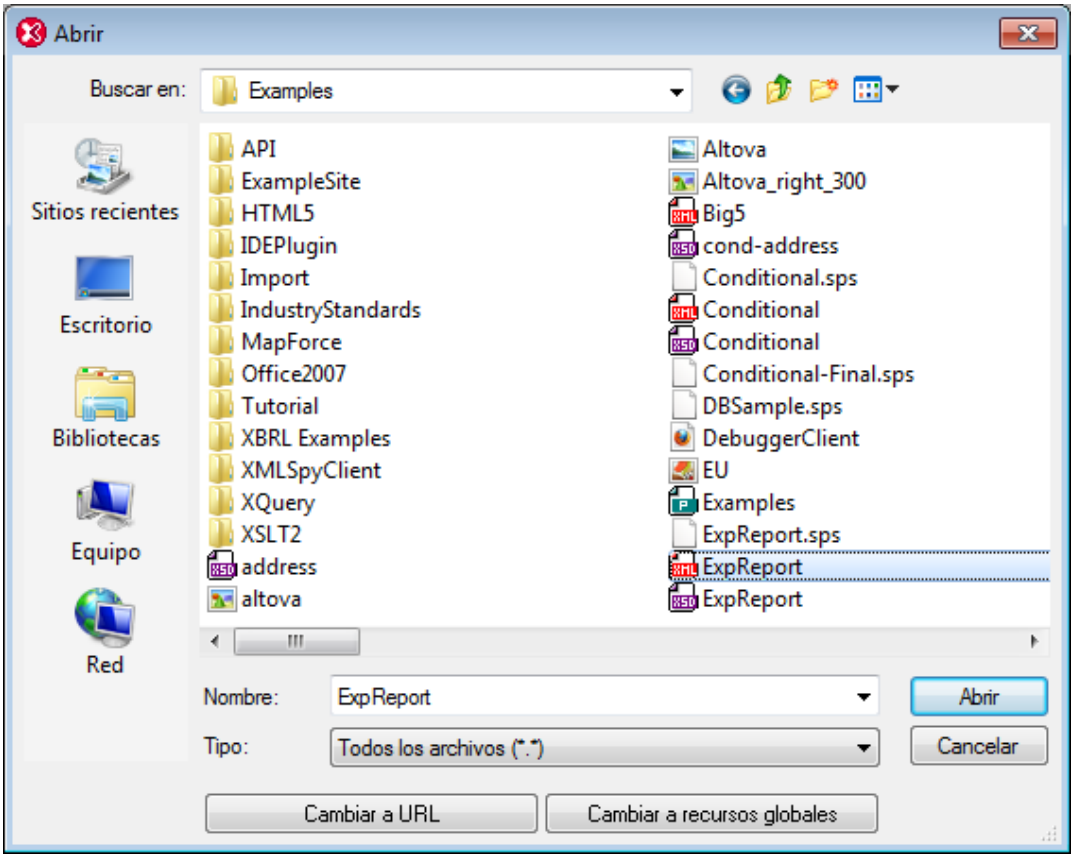
Descripción

El comando **Abrir** abre el cuadro de diálogo "Abrir" de Windows, donde puede abrir cualquier documento XML o de texto. En el cuadro de diálogo "Abrir" puede seleccionar más de un archivo. Use el cuadro combinado *Tipo:* para filtrar la lista de archivos que aparecen en el cuadro de diálogo. (La lista de tipos de archivos disponibles se puede configurar en la pestaña *Tipos de archivo* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**)). Al abrir un archivo XML, Authentic Desktop comprueba inmediatamente si tiene un formato XML correcto. Si no tiene un formato XML correcto, entonces se emite un mensaje de error. Corrija el error y seleccione el comando de menú **XML | Comprobar formato XML (F7)** para volver a revisar el documento. Si seleccionó la opción de validación automática y el archivo no es válido, entonces aparece un mensaje de error. Corrija el error y seleccione el comando **XML | Validar el documento XML (F8)** para volver a validar el archivo.

Seleccionar y guardar archivos por medio de una URL o mediante un recurso global

En varios cuadros de diálogos "Abrir archivo" y "Guardar archivo" puede seleccionar o guardar el archivo en una URL o mediante un recurso global (*imagen siguiente*). Haga clic en el botón **Cambiar a URL** o **Cambiar a recursos globales** para comenzar el proceso de selección deseado.

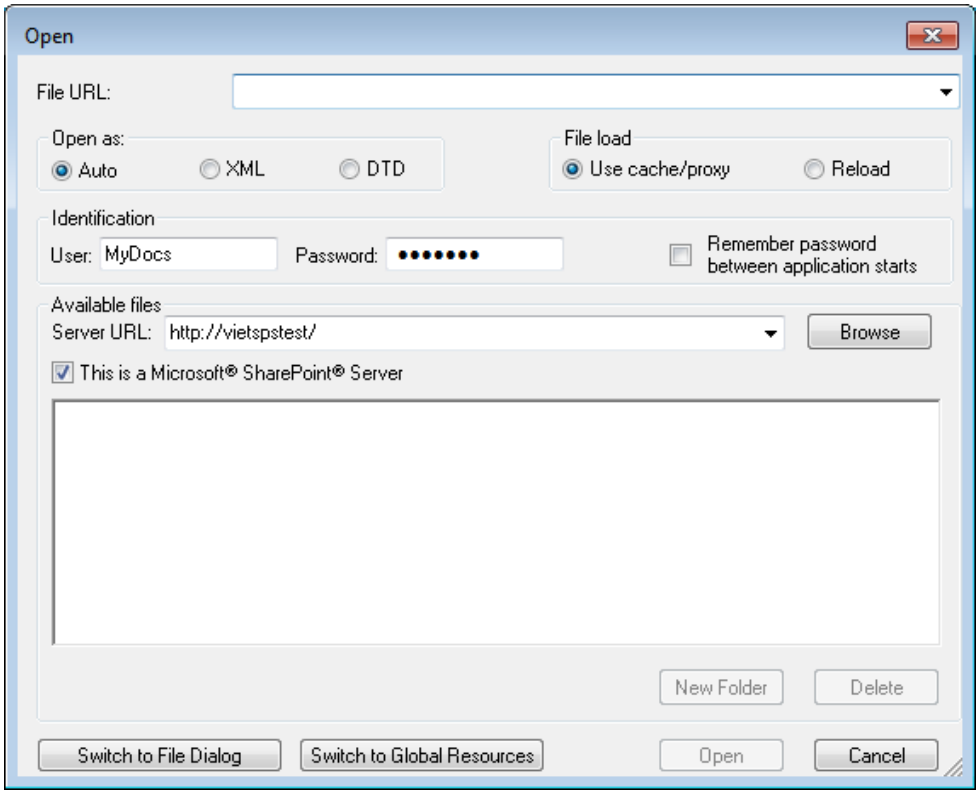
© 2015 Altova GmbH



Seleccionar archivos por medio de una URL

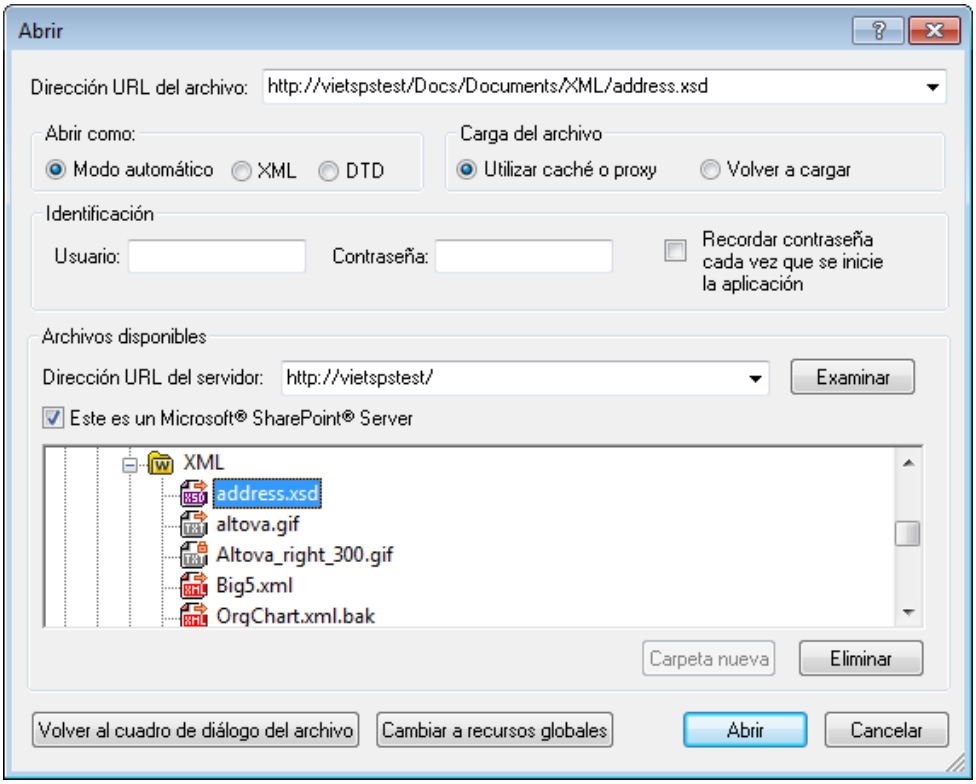
Siga estos pasos para seleccionar un archivo por medio de una URL:

1. Haga clic sobre el comando **Cambiar a URL**. Esto cambia al modo URL del cuadro de diálogo "Abrir" o "Guardar" (*el siguiente cuadro de diálogo le muestra el cuadro de diálogo "Abrir"*)



2. En el campo *Dirección URL del servidor* escriba la URL a la que desea acceder (*imagen siguiente*). Si el servidor es un servidor Microsoft® SharePoint® Server, marque la casilla *Microsoft® SharePoint® Server*. Consulte las notas sobre los servidores Microsoft® SharePoint® Server que aparecen más abajo para obtener más información sobre este tipo de servidores.
3. Si el servidor está protegido con contraseña, escriba su ID de usuario y su contraseña en los campos *Usuario* y *Contraseña*.
4. Haga clic en **Examinar** para navegar por la estructura de directorios del servidor.
5. Busque el archivo que desea cargar y haga clic en él.

© 2015 Altova GmbH



La URL del archivo aparece en el campo *Dirección URL del archivo* (*imagen anterior*). En este momento se habilita el botón **Abrir**.

6. Haga clic en el botón **Abrir** para cargar el archivo. El archivo aparece en la ventana principal.

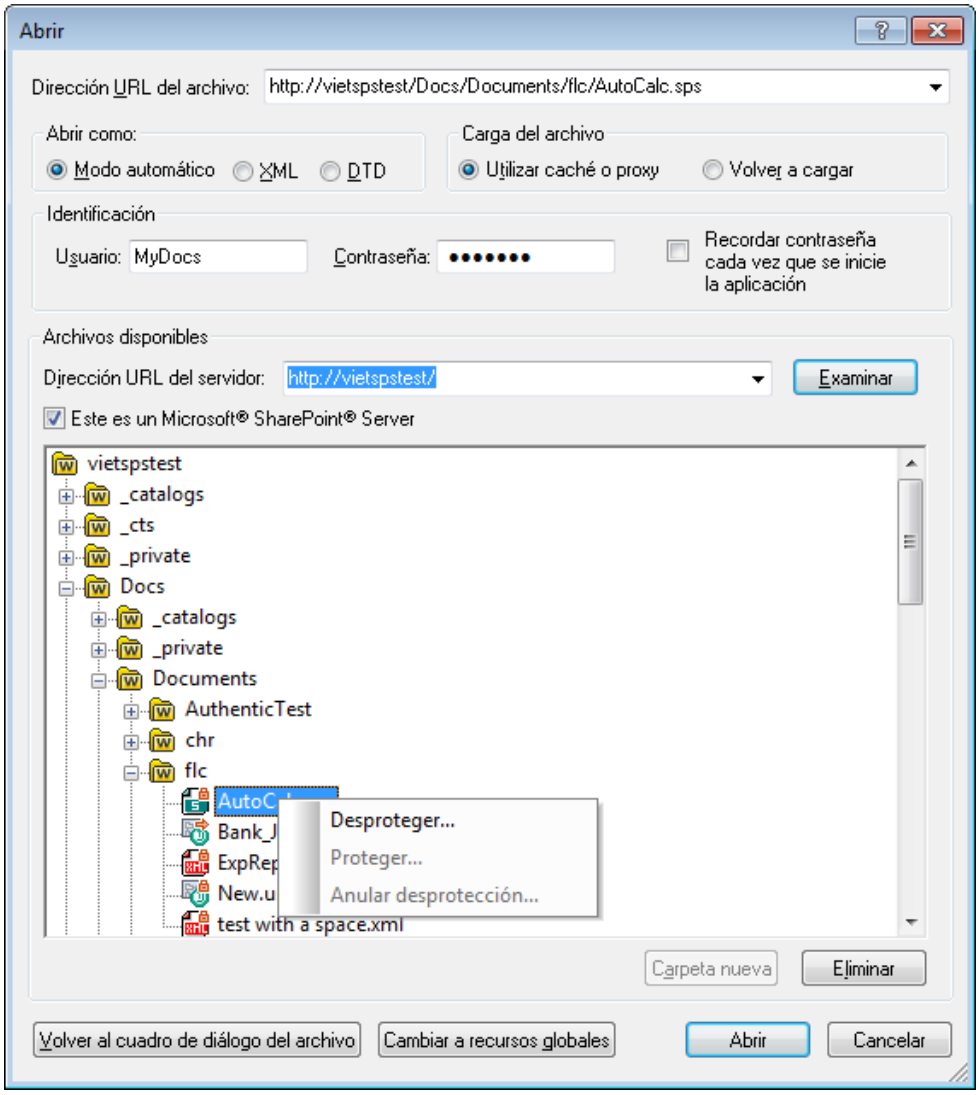
Notas:

- El botón **Examinar** sólo se habilita en servidores WebDAV y Microsoft SharePoint. Los protocolos compatibles son FTP, HTTP y HTTPS.
- Para un mayor control del proceso de carga, utilice el caché local o el servidor proxy para cargar el archivo (lo cual acelera mucho el proceso si el archivo se cargó con anterioridad). Si lo prefiere, también puede volver a cargar el archivo si trabaja, por ejemplo, con un sistema de publicación electrónica o de base de datos. En ese caso, seleccione la opción **Volver a cargar**.

Notas sobre servidores Microsoft® SharePoint® Server

Tenga en cuenta estos aspectos de los servidores Microsoft® SharePoint® Server:

- En la estructura de directorios que aparece en el panel *Archivos disponibles* (*imagen siguiente*), los iconos de archivo tienen símbolos que indican si estos están protegidos o desprotegidos.



Al hacer clic con el botón secundario en un archivo aparece un menú contextual (*imagen anterior*).

- Estos son los diferentes iconos de archivo posibles:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y después proteger.

- Después de desproteger un archivo puede editarlo en su aplicación de Altova y guardarlo con **Archivo | Guardar (Ctrl+S)**.
- Puede proteger el archivo editado con el menú contextual en el cuadro de diálogo "Abrir" (*imagen anterior*) o con el menú contextual que aparece al hacer clic con el botón secundario en la pestaña del archivo en la ventana principal de la aplicación (*imagen siguiente*).

 © 2015 Altova GmbH

- Cuando un archivo está desprotegido por otro usuario, no se puede desproteger.
- Cuando un archivo está desprotegido localmente (por usted), puede deshacer la desprotección con el comando **Deshacer desprotección** del menú contextual. Como resultado se devuelve el archivo al servidor sin ningún cambio.
- Si desprotege un archivo en una aplicación de Altova, no puede desprotegerlo en otra aplicación de Altova. En ese caso los comandos disponibles en la aplicación de Altova son **Proteger** y **Deshacer desprotección**.

Abrir y guardar archivos mediante un recurso global

Para abrir o guardar un archivo mediante un recurso global, haga clic en el botón **Cambiar a recursos globales**. Aparece un cuadro de diálogo donde puede seleccionar el recurso global. Este diálogo se describe en detalle en el apartado Recursos globales. Para una descripción general de los Recursos Globales, consulte el apartado Recursos globales en este manual.

10.1.3 Volver a cargar

Icono

Comando	*Icono*
Volver a cargar	

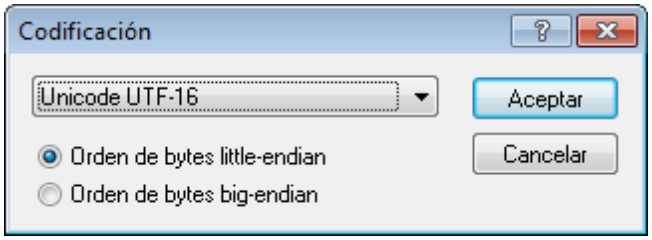
Descripción

Con el comando **Volver a cargar** puede volver a cargar los documentos que están abiertos. Esto puede ser muy útil si alguno de los documentos que están abiertos se modificó fuera de Authentic Desktop. Si se realiza algún cambio, Authentic Desktop le pregunta si desea volver a cargar el archivo. Si lo vuelve a cargar, los cambios realizados hasta entonces se perderán. Esta opción se puede cambiar en el cuadro de diálogo "Opciones" (**Herramientas | Opciones**).

© 2015 Altova GmbH

10.1.4 Codificación

Con el comando **Codificación** puede (i) ver la codificación actual del documento activo (tanto XML como no XML) y (ii) seleccionar una codificación distinta. El documento activo se guardará con esa codificación la próxima vez que se guarde.



Si cambia la codificación de un documento XML, el atributo de codificación de la declaración XML se ajusta también. Para las codificaciones de caracteres de dos y cuatro bytes (UTF-16, UCS-2 y UCS-4) también puede especificar el orden de bytes que se debe usar para el archivo. Otra manera de cambiar la codificación de un documento XML es editar el atributo `encoding` de la declaración XML del documento.

Para configurar la codificación predeterminada de sus documentos (tanto XML como no XML) utilice la pestaña *Codificación* del cuadro de diálogo "Opciones".

Nota: al guardar un documento, Authentic Desktop revisa inmediatamente la especificación de codificación y muestra el cuadro de diálogo "Codificación", donde el usuario puede seleccionar la codificación adecuada. Si al guardar el documento se detecta que contiene caracteres que no se pueden representar en la codificación seleccionada, Authentic Desktop emite una advertencia a tal efecto.

10.1.5 Cerrar, Cerrar todos, Cerrar documentos inactivos

Cerrar

El comando **Cerrar** cierra el documento activo. Si se realizaron cambios en el archivo (si hay un asterisco * después del nombre del archivo en la barra de título), Authentic Desktop le pregunta si desea guardar el archivo antes de cerrarlo.

Cerrar todos

El comando **Cerrar todos** cierra todos los documentos que están abiertos. Si se realizaron cambios en algún archivo (si hay un asterisco * después del nombre del archivo en la barra de título), Authentic Desktop le pregunta si desea guardar el archivo antes de cerrarlo.

Cerrar documentos inactivos

El comando **Cerrar documentos inactivos** cierra todos los documentos que están abiertos, excepto el documento que está activo en la ventana principal. Si se realizaron cambios en algún archivo (si hay un asterisco * después del nombre del archivo en la barra de título), Authentic Desktop le pregunta si desea guardar el archivo antes de cerrarlo.

© 2015 Altova GmbH

10.1.6 Guardar, Guardar como, Guardar todos

Iconos y acceso rápido

Comando	*Icono*	*Acceso rápido*
Guardar		**Ctrl+S**
Guardar todos		

Guardar

Este comando guarda el contenido del documento activo en el archivo desde el que se abrió. Al guardar un documento, se comprueba automáticamente si el documento tiene un formato XML correcto. El archivo también se valida automáticamente si esta opción está seleccionada en la pestaña *Archivo* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**). Authentic Desktop también revisa la declaración XML y la especificación de codificación y esta codificación se aplica al documento al guardarlo.

Guardar como

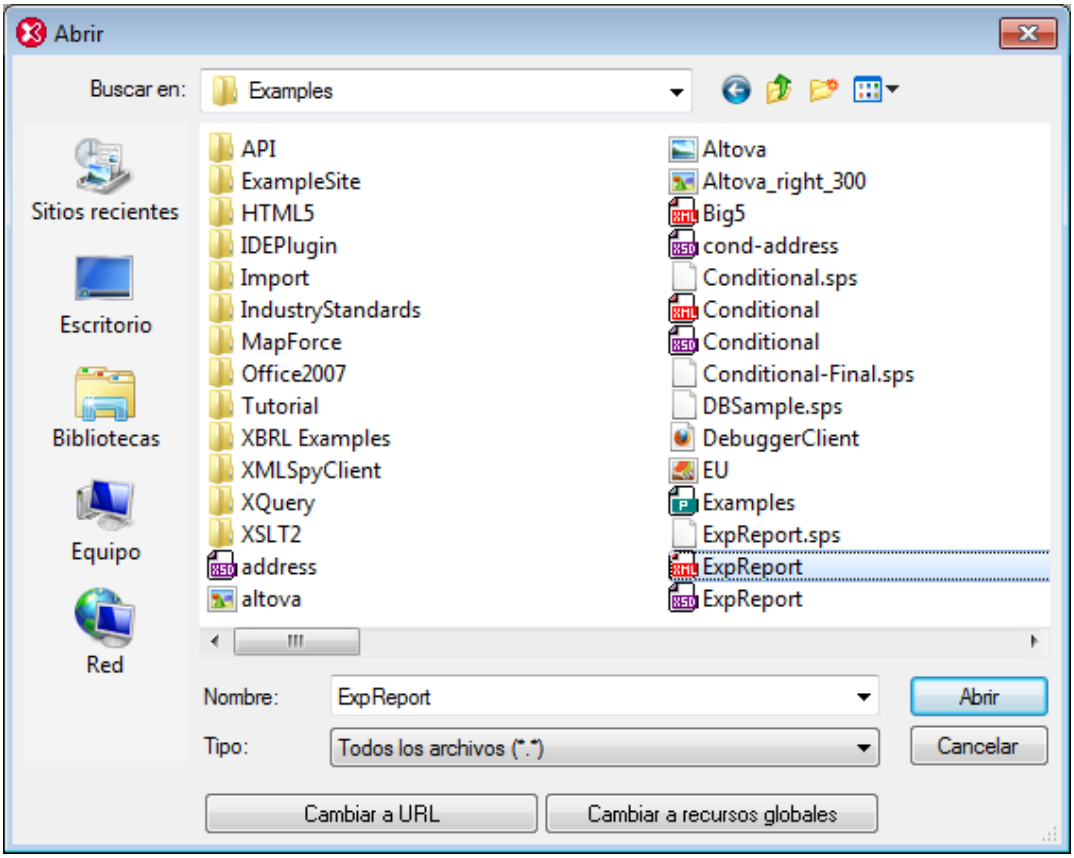
El comando **Guardar como** abre el cuadro de diálogo "Guardar como" de Windows. En este cuadro de diálogo puede introducir el nombre y la ubicación con los que desea guardar el archivo activo. Al igual que ocurre con el comando **Guardar**, la aplicación comprueba el formato XML, revisa la codificación y valida el documento.

Guardar todos

El comando **Guardar todos** guarda todos los cambios realizados en los documentos que están abiertos. Este comando es práctico si está editando varios documentos a la vez. Si alguno de los documentos no se guardó todavía (porque se acaba de crear, por ejemplo) entonces aparece el cuadro de diálogo "Guardar como".

Seleccionar y guardar archivos por medio de una URL o mediante un recurso global

En varios cuadros de diálogos "Abrir archivo" y "Guardar archivo" puede seleccionar o guardar el archivo en una URL o mediante un recurso global (*imagen siguiente*). Haga clic en el botón **Cambiar a URL** o **Cambiar a recursos globales** para comenzar el proceso de selección deseado.

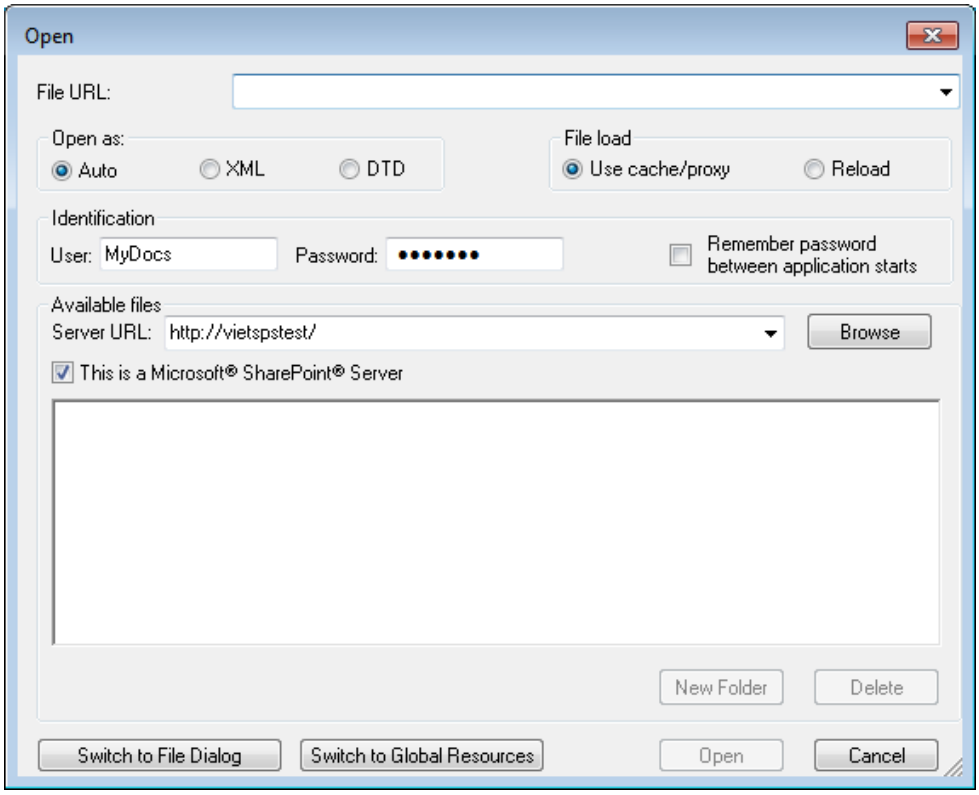


Seleccionar archivos por medio de una URL

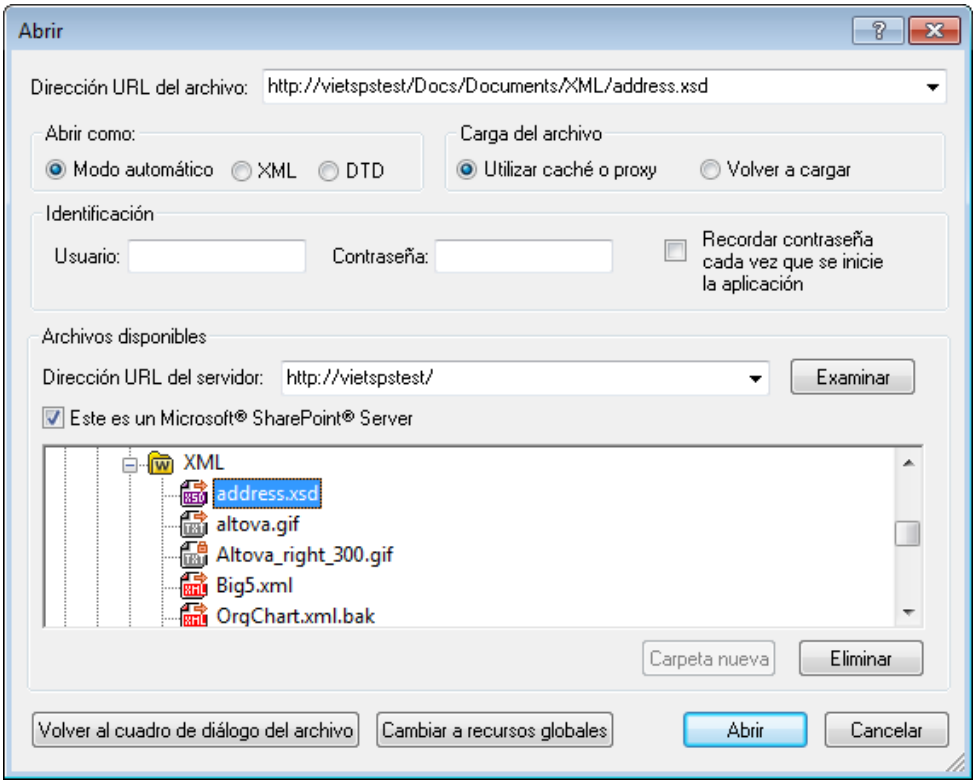
Siga estos pasos para seleccionar un archivo por medio de una URL:

1. Haga clic sobre el comando **Cambiar a URL**. Esto cambia al modo URL del cuadro de diálogo "Abrir" o "Guardar" (*el siguiente cuadro de diálogo le muestra el cuadro de diálogo "Abrir"*)

© 2015 Altova GmbH



2. En el campo *Dirección URL del servidor* escriba la URL a la que desea acceder (*imagen siguiente*). Si el servidor es un servidor Microsoft® SharePoint® Server, marque la casilla *Microsoft® SharePoint® Server*. Consulte las notas sobre los servidores Microsoft® SharePoint® Server que aparecen más abajo para obtener más información sobre este tipo de servidores.
3. Si el servidor está protegido con contraseña, escriba su ID de usuario y su contraseña en los campos *Usuario* y *Contraseña*.
4. Haga clic en **Examinar** para navegar por la estructura de directorios del servidor.
5. Busque el archivo que desea cargar y haga clic en él.



La URL del archivo aparece en el campo *Dirección URL del archivo* (*imagen anterior*). En este momento se habilita el botón **Abrir**.

6. Haga clic en el botón **Abrir** para cargar el archivo. El archivo aparece en la ventana principal.

Notas:

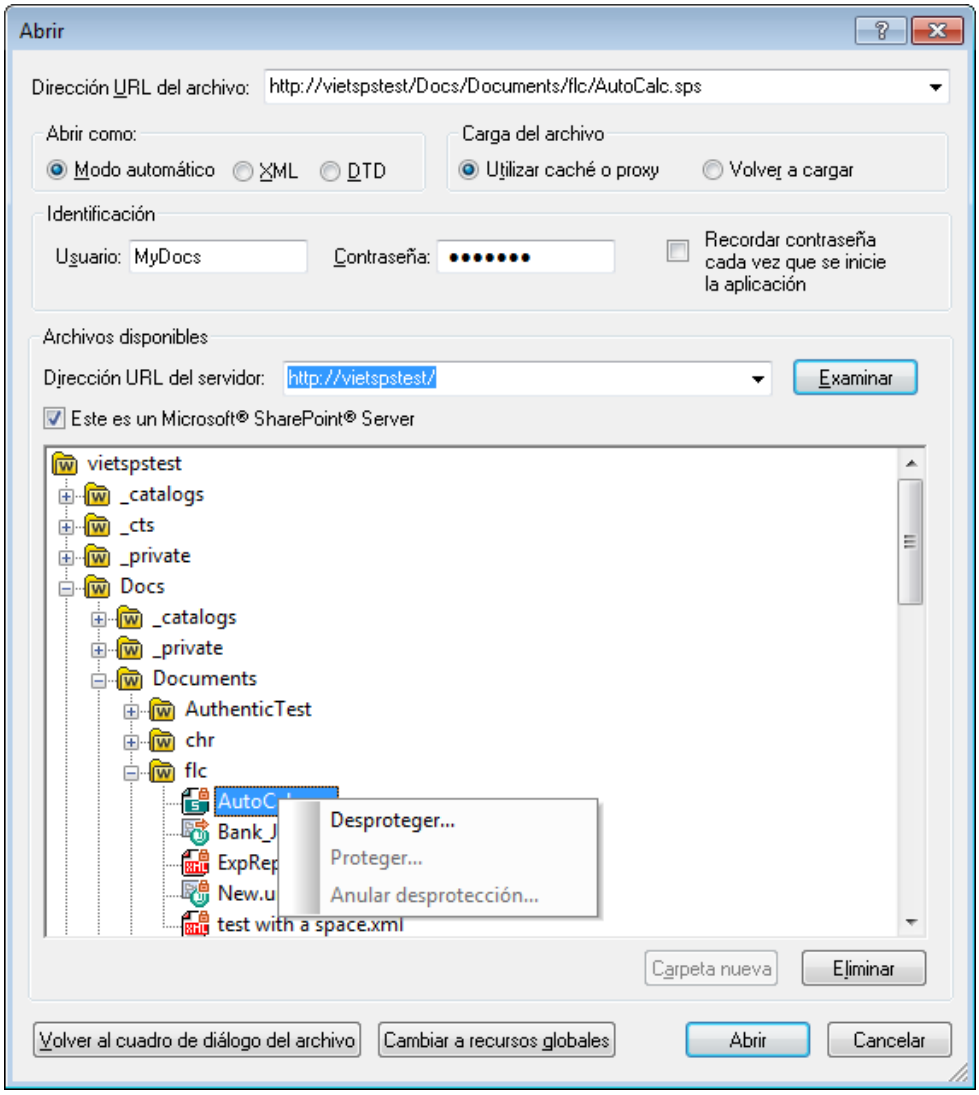
- El botón **Examinar** sólo se habilita en servidores WebDAV y Microsoft SharePoint. Los protocolos compatibles son FTP, HTTP y HTTPS.
- Para un mayor control del proceso de carga, utilice el caché local o el servidor proxy para cargar el archivo (lo cual acelera mucho el proceso si el archivo se cargó con anterioridad). Si lo prefiere, también puede volver a cargar el archivo si trabaja, por ejemplo, con un sistema de publicación electrónica o de base de datos. En ese caso, seleccione la opción **Volver a cargar**.

Notas sobre servidores Microsoft® SharePoint® Server

Tenga en cuenta estos aspectos de los servidores Microsoft® SharePoint® Server:

- En la estructura de directorios que aparece en el panel *Archivos disponibles* (*imagen siguiente*), los iconos de archivo tienen símbolos que indican si estos están protegidos o desprotegidos.

© 2015 Altova GmbH



Al hacer clic con el botón secundario en un archivo aparece un menú contextual (*imagen anterior*).

- Estos son los diferentes iconos de archivo posibles:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y después proteger.

- Después de desproteger un archivo puede editarlo en su aplicación de Altova y guardarlo con **Archivo | Guardar (Ctrl+S)**.
- Puede proteger el archivo editado con el menú contextual en el cuadro de diálogo "Abrir" (*imagen anterior*) o con el menú contextual que aparece al hacer clic con el botón secundario en la pestaña del archivo en la ventana principal de la aplicación (*imagen siguiente*).

- Cuando un archivo está desprotegido por otro usuario, no se puede desproteger.
- Cuando un archivo está desprotegido localmente (por usted), puede deshacer la desprotección con el comando **Deshacer desprotección** del menú contextual. Como resultado se devuelve el archivo al servidor sin ningún cambio.
- Si desprotege un archivo en una aplicación de Altova, no puede desprotegerlo en otra aplicación de Altova. En ese caso los comandos disponibles en la aplicación de Altova son **Proteger** y **Deshacer desprotección**.

Abrir y guardar archivos mediante un recurso global

Para abrir o guardar un archivo mediante un recurso global, haga clic en el botón **Cambiar a recursos globales**. Aparece un cuadro de diálogo donde puede seleccionar el recurso global. Este diálogo se describe en detalle en el apartado Recursos globales. Para una descripción general de los Recursos Globales, consulte el apartado Recursos globales en este manual.

© 2015 Altova GmbH

10.1.7 Enviar por correo electrónico

Icono

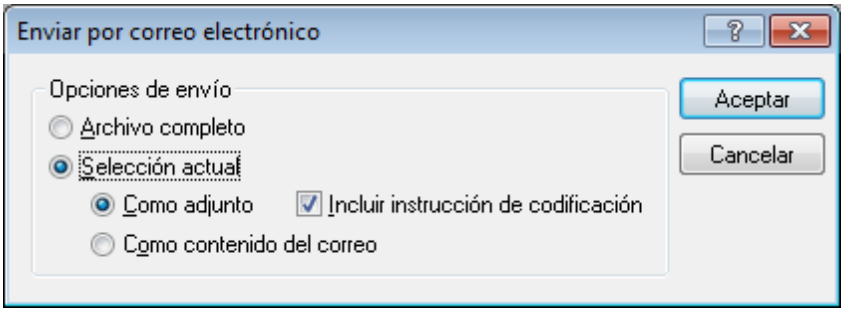
Comando	*Icono*
Enviar por correo electrónico	

Descripción

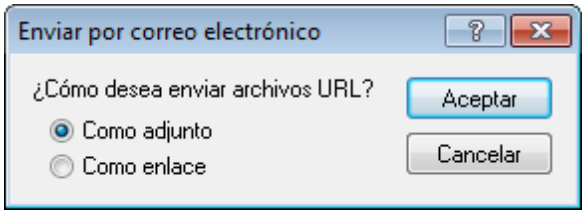
El comando **Enviar por correo electrónico...** sirve para enviar documentos XML o una selección de un documento XML por correo electrónico. El documento o la selección se puede enviar como documento adjunto, como contenido del correo o como enlace, dependiendo del tipo de documento que sea (*ver tabla siguiente*).

¿Qué puedo enviar?	¿Cómo?
Documento XML activo	Como adjunto
Selección en el documento XML activo	Como adjunto o como contenido del correo
Uno o varios archivos de la ventana Proyecto	Como adjunto
Una o varias URL de la ventana Proyecto	Como adjunto o como enlace en el correo

Cuando se invoca el comando **Enviar por correo electrónico...** para una selección del documento XML activo, aparece el cuadro de diálogo "Enviar por correo electrónico", donde puede elegir entre varias opciones (*imagen siguiente*). Si no hay texto seleccionado en el documento activo e invoca el comando **Enviar por correo electrónico...** entonces la única opción disponible en el cuadro de diálogo es la opción *Archivo completo* (las demás opciones se deshabilitan).



Puesto que los archivos enviados desde la ventana Proyecto siempre se envían como documentos adjuntos, este cuadro de diálogo no aparece y se abre directamente el correo electrónico que incluye los archivos seleccionados en forma de adjuntos. Las URL de la ventana Proyecto se pueden enviar como adjuntos o como enlaces (*imagen siguiente*). Seleccione una opción y haga clic en **Aceptar**.



© 2015 Altova GmbH

10.1.8 Imprimir

Icono y acceso rápido

Comando	*Icono*	*Acceso rápido*
Imprimir		**Ctrl+P**

Descripción

El comando **Imprimir** abre el cuadro de diálogo "Imprimir", donde puede configurar la impresión. Después puede imprimir el documento activo tal y como aparece en la vista actual.

10.1.9 Vista previa de impresión, Configurar impresión

Vista previa de impresión

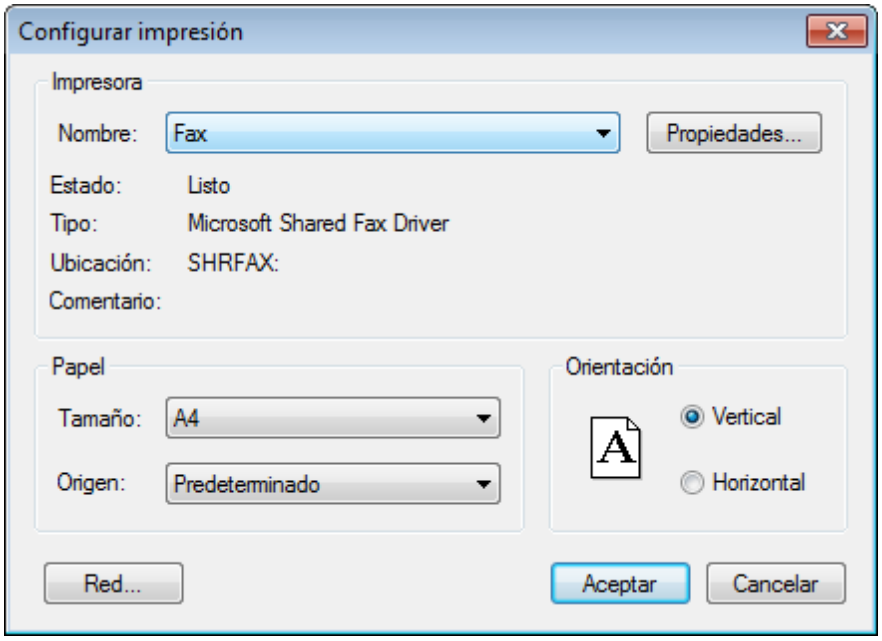
Al hacer clic en el comando **Vista previa de impresión** de la vista Texto y Explorador se abre una vista previa de impresión del documento activo.

El modo *Vista previa de impresión* incluye una barra de herramientas (en la parte superior izquierda) con opciones relacionadas con la impresión y con la vista previa. Puede agrandar o reducir la vista previa con los botones **Acercarse** y **Alejarse** de la barra de herramientas. Si después de pulsar el botón **Acercarse** la vista previa cabe en una sola página, el botón **Una página / Dos páginas** se deshabilita. Si la vista previa ocupa más de una página, puede usar los botones **Siguiente** y **Anterior** para navegar por la vista previa. La barra de herramientas también incluye un botón **Imprimir** y otro para **Cerrar** la ventana de vista previa.

Nota: siga estos pasos para habilitar el uso de colores e imágenes de fondo en la vista previa de impresión: (i) en el menú **Herramientas** de Internet Explorer, haga clic en **Opciones de Internet** y después en la pestaña *Opciones avanzadas*. (ii) En el panel Configuración, marque la casilla *Imprimir colores e imágenes de fondo* de la sección Imprimir. (iii) Haga clic en **Aceptar** para terminar.

Configurar impresión

El comando **Configurar impresión** abre el cuadro de diálogo "Configurar impresión" (*imagen siguiente*), donde puede seleccionar el formato del papel y la orientación de la página, entre otras opciones. Estas opciones se aplican a todos los trabajos de impresión subsiguientes.



© 2015 Altova GmbH

10.1.10 Archivos recientes, Salir

Archivos recientes

El menú **Archivo** incluye una lista con los últimos 9 archivos que se abrieron en Authentic Desktop. El primer archivo de esta lista es el último archivo que se abrió. Para abrir un archivo de la lista, haga clic en su nombre. Para abrir un archivo de la lista usando el teclado, pulse **Alt+A** para abrir el menú **Archivo** y después pulse el número del archivo que desea abrir.

Salir

El comando **Salir** sirve para cerrar Authentic Desktop. Si hay algún archivo abierto que no se guardó, se le pedirá que guarde los cambios. Authentic Desktop también guarda los cambios realizados en las opciones de configuración del programa y la información sobre archivos recientes.

10.2 Menú Edición

Los comandos del menú **Edición** sirven para editar documentos en Authentic Desktop. Algunos ejemplos son los comandos **Deshacer**, **Rehacer**, **Cortar**, **Copar**, **Pegar**, **Eliminar**, **Seleccionar todo**, **Buscar**, **Buscar siguiente** y **Reemplazar**.

© 2015 Altova GmbH

10.2.1 Deshacer, Rehacer

Iconos y acceso rápido

Comando	*Icono*	*Acceso rápido*
Deshacer		**Ctrl+Z**
Rehacer		**Ctrl+Y**

Deshacer

El comando **Deshacer** permite eliminar todos los cambios realizados y devolver el archivo a versiones anteriores. Todos los cambios se pueden deshacer uno por uno y no hay un límite de operaciones deshacer. El historial de operaciones deshacer se mantiene tras usar el comando **Guardar**, lo cual permite devolver el archivo al estado en que estaba antes de que guardara los cambios. Es decir puede recorrer el historial de acciones con los comandos **Deshacer** y **Rehacer** (*ver siguiente*).

Rehacer

El comando **Rehacer** permite rehacer las acciones que deshizo con el comando **Deshacer**. Esto significa que puede ir adelante y atrás en el historial de acciones con los comandos **Deshacer** y **Rehacer**.

10.2.2 Cortar, Copiar, Pegar, Eliminar

Iconos y acceso rápido

Comando	*Icono*	*Acceso rápido*
Cortar		**Ctrl+X** o **Mayús+Supr**
Copiar		**Ctrl+C**
Pegar		**Ctrl+V**
Eliminar		**Supr**

Cortar

El comando **Cortar** copia el texto o los componentes seleccionados en el portapapeles y los elimina de su ubicación actual.

Copiar

El comando **Copiar** copia el texto o los componentes seleccionados en el portapapeles. Puede usar este comando para duplicar datos dentro de Authentic Desktop o para mover datos a otra aplicación.

Pegar

El comando **Pegar** inserta el contenido del portapapeles en la posición actual del cursor.

Eliminar

El comando **Eliminar** elimina el texto o los componentes seleccionados sin colocarlos en el portapapeles.

© 2015 Altova GmbH

10.2.3 Seleccionar todo

El comando **Seleccionar todo** (**Ctrl+A**) sirve para seleccionar el contenido de todo el documento.

10.2.4 Buscar, Buscar siguiente

Iconos y acceso rápido

Comando	*Icono*	*Acceso rápido*
Buscar		**Ctrl+F**
Buscar siguiente		**F3**

Buscar

El comando **Buscar** abre el cuadro de diálogo "Buscar", donde puede indicar qué cadena de texto desea buscar y configurar la búsqueda. Escriba el término de búsqueda en el cuadro de texto *Buscar* o seleccione una opción del cuadro combinado, que recuerda los 10 términos de búsqueda más recientes. Después configure las opciones de búsqueda.

Los comandos **Buscar** y **Buscar siguiente** también sirven para buscar nombres de archivo y carpeta cuando en la ventana Proyecto está seleccionado un proyecto.

Buscar siguiente

El comando **Buscar siguiente** repite la última búsqueda realizada con el comando **Buscar** y busca la siguiente instancia del término de búsqueda.

© 2015 Altova GmbH

10.2.5 Reemplazar

Icono y acceso rápido

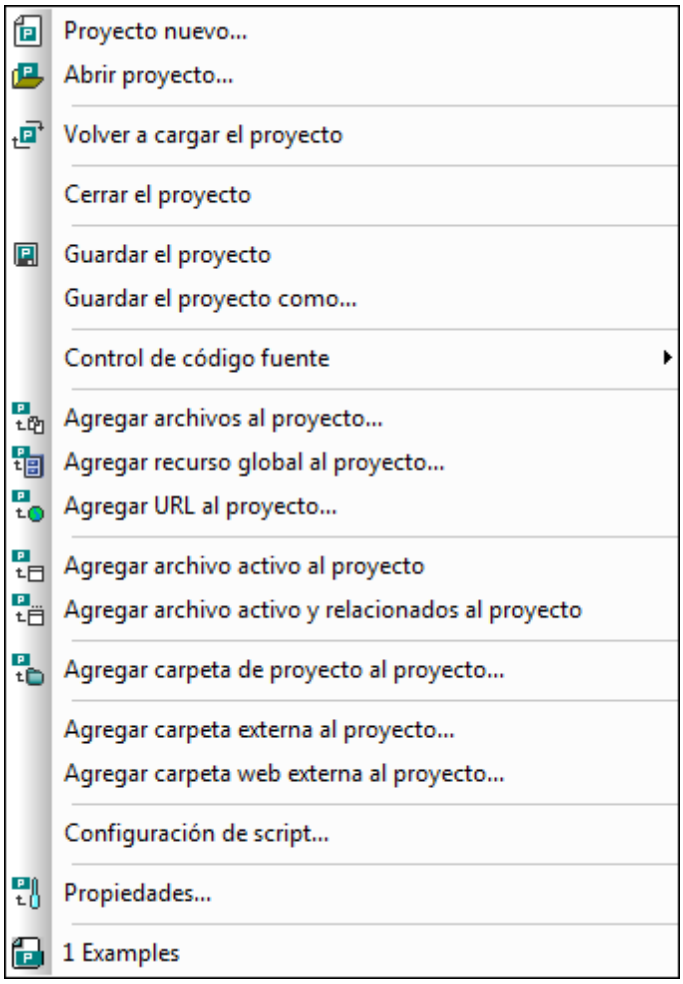
Comando	*Icono*	*Acceso rápido*
Reemplazar		**Ctrl+H**

Descripción

Con el comando **Reemplazar** puede buscar y reemplazar una cadena de texto con otra. Este comando tiene las mismas características que el comando **Buscar**. Puede reemplazar los términos encontrados uno por uno o usar el botón **Reemplazar todos** para realizar una operación global de búsqueda y reemplazo.

10.3 Menú Proyecto

Authentic Desktop usa una estructura en forma de árbol para gestionar y organizar varios archivos o direcciones URL en proyectos XML. Los archivos y direcciones URL se pueden organizar por carpetas según su extensión de archivo o cualquier otro criterio, para que pueda estructurar su proyecto y realizar operaciones por lotes con total facilidad.



Nota importante: la mayoría de los comandos para trabajar con proyectos también están disponibles en el menú contextual que aparece al hacer clic con el botón secundario en cualquier elemento de la ventana Proyecto.

Rutas de archivo relativas y absolutas

Los proyectos se guardan como archivo de proyecto y tienen la extensión `.spp`. Se trata, de hecho, de documentos XML que se pueden editar como si fueran archivos XML normales y corrientes. En el archivo de proyecto se usan rutas de archivo absolutas para archivos y carpetas que están al mismo nivel o en un nivel superior, mientras que para archivos y carpetas de la carpeta actual o subcarpetas se usan rutas de archivo relativas. Por ejemplo, si la estructura de directorios de su proyecto es como esta:

```
|-Carpeta1
|      |
```


© 2015 Altova GmbH

```
|      |-Carpeta2
|            |
|            |-Carpeta3
|                  |
|                  |-Carpeta4
```

Y su archivo `.spp` está guardado en la `Carpeta3`, entonces las referencias a los archivos de la `Carpeta1` y `Carpeta2` serán así::

```
c:\Carpeta1\NombreDeArchivo.ext
c:\Carpeta1\Carpeta2\NombreDeArchivo.ext
```

Por su parte, las referencias a los archivos de la `Carpeta3` y `Carpeta4` serán así:

```
.\NombreDeArchivo.ext
.\Carpeta4\NombreDeArchivo.ext
```

Si desea que todos las rutas de archivo sean relativas, guarde los archivos `.spp` en el directorio raíz de su disco duro.

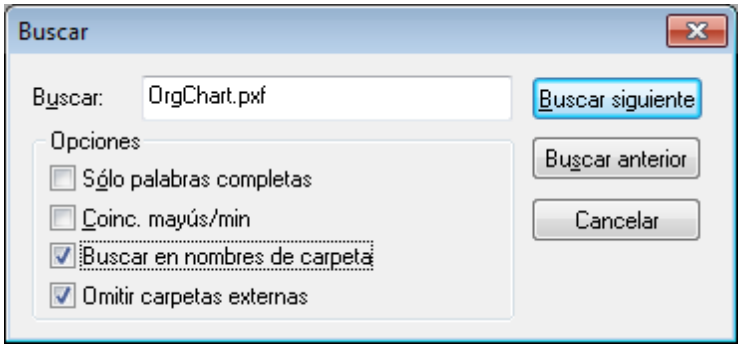
Arrastrar y colocar

En la ventana Proyecto puede arrastrar carpetas hasta otra carpeta o hasta otra posición de la misma carpeta. Los archivos se pueden arrastrar hasta otra carpeta, pero no se pueden cambiar de posición dentro de la misma carpeta (en las carpetas los archivos se organizan alfabéticamente). Además, puede arrastrar archivos y carpetas desde el explorador de Windows hasta la ventana Proyecto de XMLSpy.

Cómo realizar búsquedas en el proyecto

Puede buscar archivos y carpetas del proyecto por su nombre o sólo parte del nombre. Si se encuentra un archivo o una carpeta que coincida con la búsqueda, estos se resaltan uno por uno en la ventana Proyecto.

Para iniciar una búsqueda, en la ventana Proyecto seleccione la carpeta de proyecto en la que desea buscar y después seleccione el comando **Edición | Buscar** (**Ctrl+F**). Aparece el cuadro de diálogo Buscar (*imagen siguiente*). Escriba el texto que desea buscar y seleccione las opciones de búsqueda que correspondan (*véase más adelante*).



Este cuadro de diálogo ofrece estas opciones de búsqueda:

- Puede buscar *palabras completas* solamente. Esto limita los resultados porque el nombre de carpeta o de archivo debe incluir la palabra completa. En los nombres de

archivo, la parte situada delante y detrás del punto (sin el punto) se considera una palabra.
- Puede buscar sólo los archivos o carpetas cuyo nombre coincida con el texto de búsqueda en el uso de *mayúsculas y minúsculas*.
- Puede buscar en los *nombres de carpeta* o sólo en los nombres de archivo.
- Puede incluir o excluir las *carpetas externas* en la búsqueda. Las carpetas externas son carpetas reales del sistema de archivos o de la red. Por el contrario, las carpetas de proyecto se crean dentro del proyecto pero no en el sistema.

Si la búsqueda encuentra resultados, se resalta el primer resultado en la ventana Proyecto. Para navegar por el resto de resultados de la búsqueda, haga clic en **Buscar siguiente** y **Buscar anterior**.

Actualizar los proyectos

Si se realizan cambios en una carpeta externa, el cambio no se refleja en la ventana Proyecto a no ser que se actualice el proyecto.

Recursos globales en el menú contextual

Al hacer clic con el botón secundario en una carpeta en la ventana Proyecto, el menú contextual que aparece incluye la opción **Agregar recurso global**, que sirve para añadir recursos globales. Al hacer clic en este comando aparece el cuadro de diálogo "Agregar recurso global". Este diálogo enumera todos los recursos globales de tipo archivo o carpeta disponibles en el archivo XML de recursos globales activo. Seleccione un recurso global y este se añade a la carpeta de proyecto seleccionada.

Proyectos y proveedores de control de versiones

Si desea añadir un proyecto de Authentic Desktop a un repositorio de control de versiones, antes es necesario comprobar si la posición de los archivos del proyecto en la estructura del sistema de archivos permite añadir archivos desde un nivel inferior (entendiendo el directorio raíz como nivel superior de la estructura de directorios).

En otras palabras, el directorio donde está ubicado el **archivo de proyecto** básicamente representa el **directorio raíz** del proyecto en el repositorio de control de versiones. Los archivos que se añadan desde un nivel superior (al directorio raíz) se añaden al proyecto de Authentic Desktop, pero su posición en el repositorio puede ser diferente a la esperada, si es que se pueden colocar en esa posición.

Por ejemplo, en la estructura de carpetas que aparece más arriba, si un archivo de proyecto se guarda en la `Carpeta3` y se pone bajo control de versiones:

- Los archivos que se añadan en la `Carpeta1` no se pueden poner bajo control de versiones.
- Los archivos que se añadan a la `Carpeta2` se ponen bajo control de versiones en el directorio raíz del repositorio y no dentro de la carpeta de proyecto.
- Los archivos que se añadan a la `Carpeta3` y `Carpeta4` se añaden en la posición esperada y se ponen bajo control de versiones.

© 2015 Altova GmbH

10.3.1 Proyecto nuevo

El comando **Proyecto nuevo** crea un proyecto **nuevo** en Authentic Desktop. Si al hacer clic en este comando está trabajando en otro proyecto, aparece un mensaje preguntando si desea cerrar todos los documentos que pertenecen al proyecto actual.

10.3.2 Abrir proyecto

El comando **Abrir proyecto...** abre un proyecto de Authentic Desktop. Si al hacer clic en este comando está trabajando en otro proyecto, primero se cierra el proyecto actual.

© 2015 Altova GmbH

10.3.3 Volver a cargar el proyecto

El comando **Volver a cargar el proyecto** vuelve a cargar el proyecto desde el disco. Si trabaja en un entorno de varios usuarios, a veces puede ser necesario volver a cargar el proyecto si otro usuario realizó cambios en el proyecto.

Nota importante: los archivos de proyecto (archivos `.spp`) son de hecho documentos XML que se pueden editar como cualquier otro archivo XML.

10.3.4 Cerrar el proyecto

El comando **Cerrar el proyecto** sirve para **cerrar** el proyecto activo. Si se realizaron cambios en el proyecto, la aplicación le pregunta si desea guardar el proyecto antes de cerrarlo. Si se realizaron cambios en el proyecto, esto se indica en la ventana Proyecto con un asterisco junto al nombre del proyecto.

© 2015 Altova GmbH

10.3.5 Guardar el proyecto, Guardar el proyecto como

El comando **Guardar el proyecto** sirve para **guardar** el proyecto actual. También puede hacer clic en la ventana Proyecto y pulsar el icono .

El comando **Guardar el proyecto como** sirve para **guardar** el proyecto actual con un nombre nuevo.

10.3.6 Control de código fuente

Su aplicación de Altova es compatible con Microsoft SourceSafe y otros repositorios compatiblles. El apartado Sistemas de control de código fuente compatibles incluye una lista con todos los sistemas compatibles. Este apartado describe los comandos del submenú **Proyecto | Control de código fuente**, que sirven para trabajar con el sistema de control de código fuente desde su aplicación de Altova.

Resumen de la función de control de código fuente

Este es el mecanismo para colocar archivos de un proyecto bajo control de código fuente:

1. En Authentic Desktop se crea una carpeta de proyecto que contiene los archivos que se deben colocar bajo control de código fuente. Esta carpeta se corresponde normalmente con una carpeta local donde están guardados los archivos del proyecto. La ruta de acceso de la carpeta local se denomina ruta de acceso local.
2. En la base de datos del sistema de control de código fuente (también llamada repositorio) se crea una carpeta que contendrá los archivos que se deben poner bajo control de código fuente.
3. Los archivos del proyecto se añaden al control de código fuente con el comando **Proyecto | Control de código fuente | Agregar al control de código fuente**.
4. Las acciones de control de código fuente (como proteger, desproteger y quitar archivos del control de código fuente) se pueden llevar a cabo con los comandos del submenú **Proyecto | Control de código fuente**. Los comandos de este submenú se describen en los siguientes subapartados.

Nota: si desea cambiar de proveedor de control de código fuente, tiene dos opciones: (i) cambiar las opciones de control de código fuente (**Herramientas | Opciones | Control de código fuente**) o (ii) cambiar de proveedor en el diálogo "Cambiar control de código fuente" (**Proyecto | Control de código fuente | Cambiar control de código fuente**).

Nota: un proyecto de control de código fuente no es lo mismo que un proyecto de la aplicación. Los proyectos de control de código fuente dependen de la estructura del directorio, mientras que los proyectos de Authentic Desktop son construcciones lógicas que no dependen directamente de la estructura del directorio.

Para más información, consulte la sección Control de código fuente.

Abrir desde el control de código fuente

El comando **Abrir desde el control de código fuente** crea un proyecto de aplicación nuevo a partir de un proyecto bajo control de código fuente.

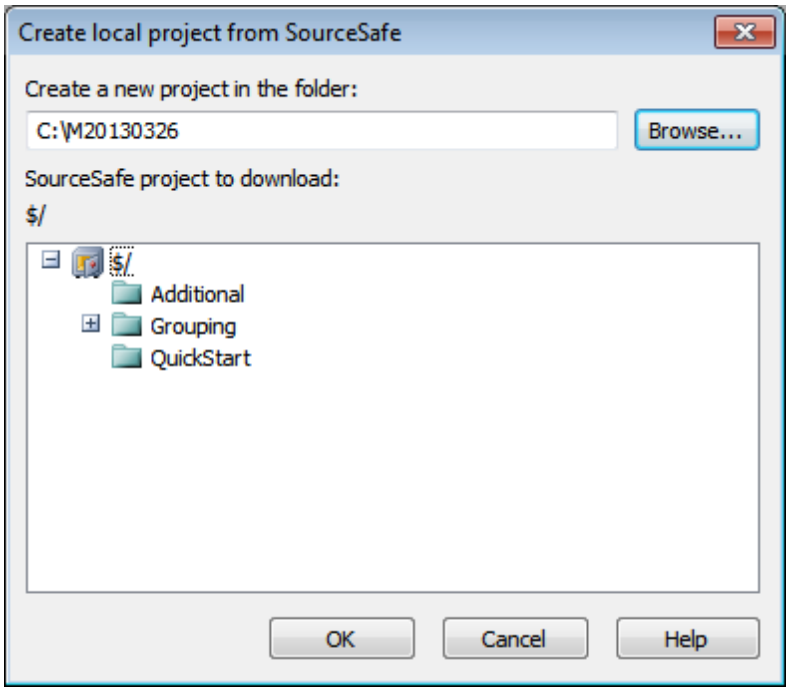
Estos son los pasos para crear el proyecto:

1. Dependiendo del sistema de control de código fuente que utilice, quizás sea necesario comprobar que ningún archivo del proyecto está desprotegido antes de crear un proyecto nuevo.
2. No es necesario tener abierto un proyecto en la aplicación.
3. Seleccione el comando **Proyecto | Control de código fuente | Abrir desde el control de código fuente**.
4. El sistema de control de código fuente que tiene instalado muestra sus cuadros de diálogo de verificación y conexión. Conéctese al repositorio pertinente (es decir, a la

© 2015 Altova GmbH

carpeta del repositorio que se corresponde con la carpeta local).

5. En el cuadro de diálogo que aparece (*imagen siguiente*) busque la carpeta local en la que desea copiar el contenido de la carpeta del repositorio (a la que se acaba de conectar). En la imagen siguiente, por ejemplo, la carpeta asociada del repositorio se llama `MiProyecto` y se representa con el signo `$`. La carpeta local es `C:\M20130326`.



6. Haga clic en **OK**. El contenido de la carpeta asociada (`MyProject`) se copiará a la carpeta local `C:\M20130326`. y aparece un cuadro de diálogo pidiendo que seleccione el archivo de proyecto (archivo `.spp`) que desea crear como proyecto nuevo.
7. Seleccione el archivo `.spp` que se copió a la carpeta local. En nuestro ejemplo, se trata del archivo `MiProyecto.spp` situado en la carpeta `C:\M20130326`. En la aplicación se crea un proyecto nuevo llamado `MiProyecto` y este proyecto aparece en la ventana Proyecto. Los archivos del proyecto están en la carpeta `C:\M20130326`.

Símbolos de control de código fuente

Las carpetas y los archivos aparecen con un icono diferente dependiendo de su estado:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Habilitar control de código fuente

El comando **Habilitar control de código fuente** sirve para habilitar/deshabilitar el control de código fuente para un proyecto de la aplicación. Este comando habilita/deshabilita el control de código fuente para todo el proyecto, aunque se seleccione desde el menú contextual de un archivo o de una carpeta. Una vez habilitado el control de código fuente, el estado de los archivos se recupera y se muestra en la ventana proyecto.

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Obtener la versión más reciente

El comando **Obtener la versión más reciente** (del menú **Proyecto | Control de código fuente**) recupera y coloca en el directorio de trabajo la versión más reciente de los archivos seleccionados del control de código fuente. Los archivos se recuperan como archivos de solo lectura y no están desprotegidos. Este comando es igual que el comando **Obtener**, la única diferencia es que no aparece el cuadro de diálogo "Obtener".

Si los archivos seleccionados están desprotegidos, la acción que se toma dependerá de lo que haga su sistema de control en situaciones así. Por lo general, el sistema de control de código fuente le preguntará si desea reemplazar el archivo desprotegido, combinarlo o dejarlo tal y como está.

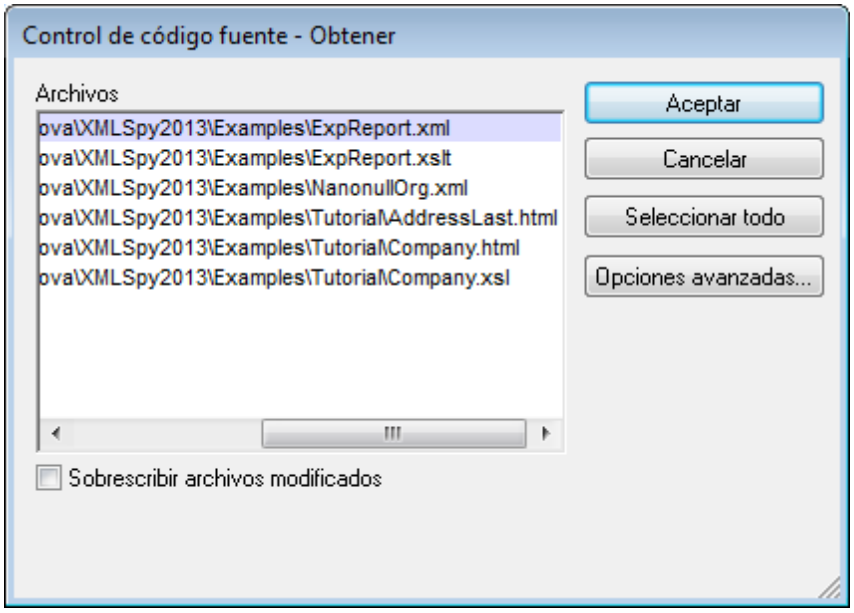
Nota: cuando seleccione este comando para una carpeta, el comando se aplica de forma recursiva. Esto significa que se aplica a todos los archivos situados dentro de la carpeta actual.

Obtener, Obtener carpetas

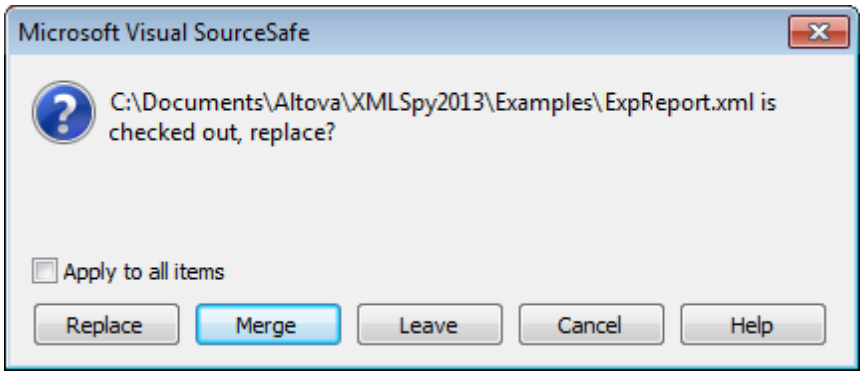
El comando **Obtener** (del menú **Proyecto | Control de código fuente**) recupera los archivos del repositorio como archivos de solo lectura. (Para poder editar un archivo, antes es necesario desprotegerlo.) El cuadro de diálogo "Obtener" enumera los archivos del objeto (proyecto o carpeta) sobre el que se ejecutó el comando **Obtener** (*imagen siguiente*). En el cuadro de diálogo puede marcar las casillas de los archivos que desea recuperar.

Nota: el comando **Obtener carpetas** sirve para seleccionar subcarpetas del repositorio (si su sistema de control de código fuente permite esta opción).

© 2015 Altova GmbH

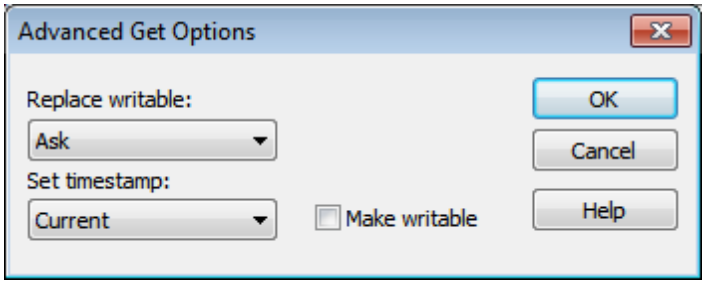


Si quiere, puede sobrescribir los archivos desprotegidos modificados marcando la opción *Sobrescribir archivos modificados* situada en la parte inferior del cuadro de diálogo. Al hacer clic en **Aceptar** los archivos se sobrescriben. Si alguno de los archivos sobrescritos está abierto, aparece un aviso preguntando si desea volver a cargar los archivos (botón **Volver a cargar**), cerrar los archivos (botón **Cerrar**) o conservar la vista actual del archivo (botón **Cancelar**).



Opciones avanzadas

El cuadro de diálogo "Opciones avanzadas" (*imagen siguiente*) se abre con el botón **Opciones avanzadas** del cuadro de diálogo "Obtener" (*primera imagen de este apartado*).



Aquí puede seleccionar (i) si reemplaza los archivos que se pueden escribir y que están desprotegidos, (ii) la marca de tiempo y (iii) si la propiedad de solo lectura del archivo recuperado se cambia para que el archivo se pueda escribir.

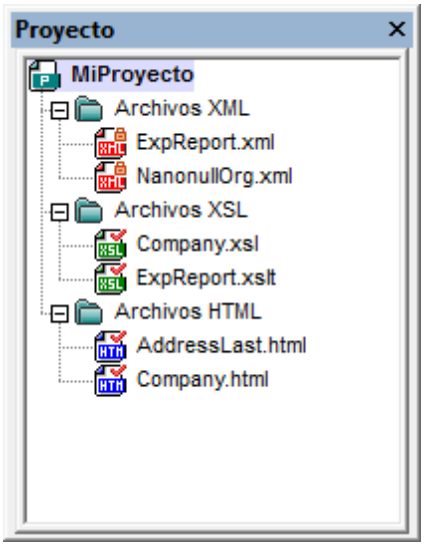
Desproteger, Proteger

Tras poner un archivo de proyecto bajo control de código fuente, puede desprotegerlo o protegerlo. Para ello seleccione el archivo (en la ventana Proyecto) y seleccione el comando pertinente en el menú **Proyecto | Control de código fuente**, es decir **Desproteger** o **Proteger**.

Cuando se desprotege un archivo, se pone una copia del archivo del repositorio en la carpeta local. Los archivos desprotegidos se pueden editar. Si un archivo que está bajo control de código fuente no se desprotege, no se puede editar. Después de editar el archivo, los cambios se pueden guardar en el repositorio protegiendo el archivo. Incluso si no guarda el archivo, al protegerlo se guardan los cambios en el repositorio. El icono del archivo indica si está desprotegido o protegido.

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Si selecciona el proyecto o una carpeta del proyecto, se seleccionan todos los archivos del objeto seleccionado. Para seleccionar varios objetos (archivos y carpetas) pulse la tecla **Ctrl** mientras hace clic en los objetos. La imagen siguiente muestra un proyecto que se desprotegió. Después se protegió el archivo `ExpReport.xml`.



Guardar cambios y rechazar cambios

Recuerde que al proteger un archivo puede elegir si se protegen los cambios o no. Es decir, puede guardar los cambios en el repositorio y seguir con el archivo desprotegido. Esto es muy útil si desea guardar cambios cada cierto tiempo y después seguir editando el archivo

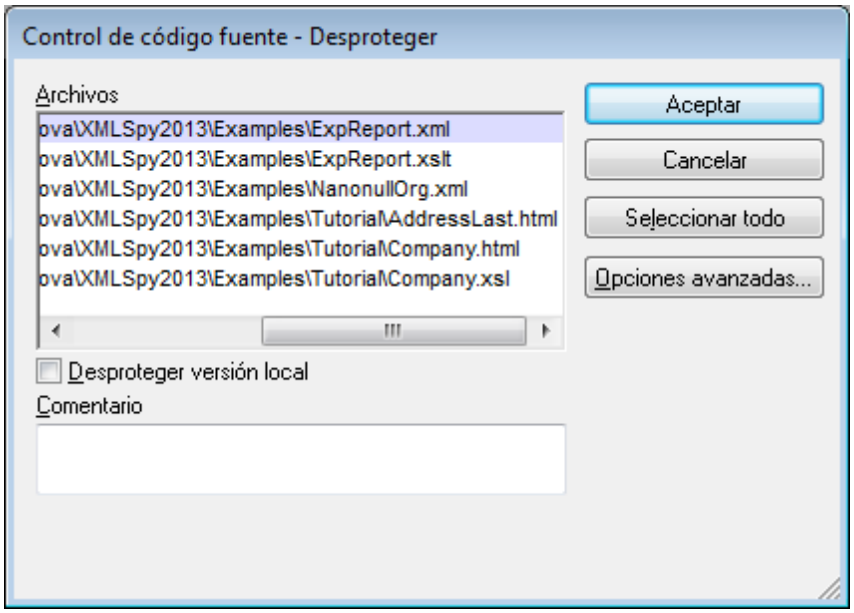
© 2015 Altova GmbH

desprotegido.

Si desprotegió un archivo y realizó cambios pero quiere rechazarlos por cualquier motivo, puede volver a la versión del documento guardada en el repositorio con solo hacer clic en el comando **Proyecto | Control de código fuente | Deshacer desprotección**.

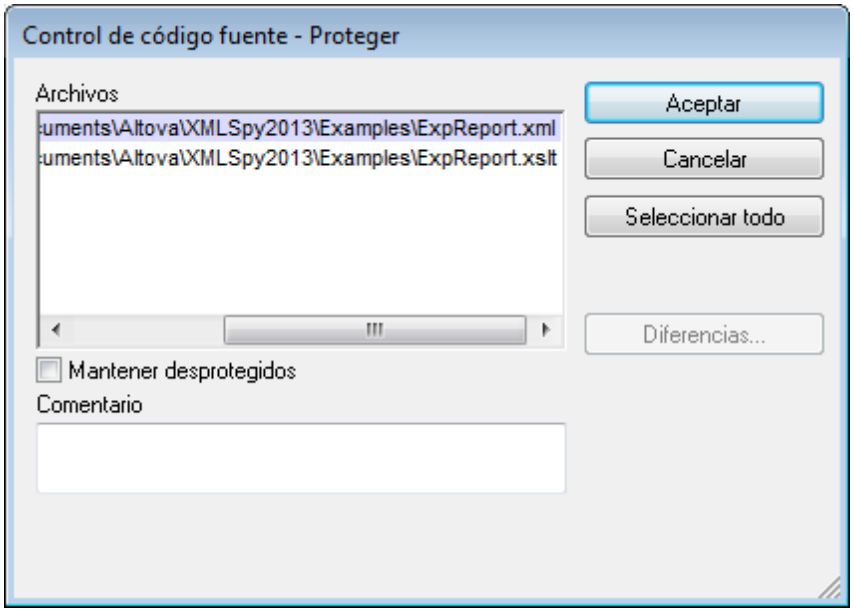
Desproteger

El cuadro de diálogo "Desproteger" (*imagen siguiente*) sirve para: (i) seleccionar los archivos que desea desproteger y (ii) seleccionar si se desprotege la versión del repositorio o la versión local.



Proteger

El cuadro de diálogo "Proteger" (*imagen siguiente*) sirve para: (i) seleccionar los archivos que desea proteger y (ii) mantener el archivo desprotegido.



Nota: en estos dos cuadros de diálogo aparecen varios archivos si el objeto seleccionado (el proyecto o las carpetas de proyecto) contienen varios archivos.

Anular desprotección

Si desprotegió un archivo y realizó cambios pero ahora prefiere rechazar esos cambios por cualquier motivo, puede volver a la versión del documento guardada en el repositorio. Para ello seleccione el comando **Proyecto | Control de código fuente | Anular desprotección**.

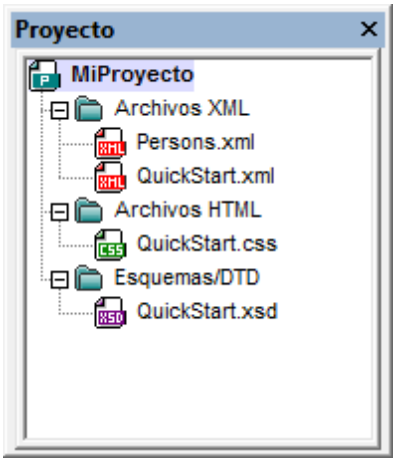
Las carpetas y los archivos aparecen con un icono diferente dependiendo de su estado:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Agregar al control de código fuente

Después de añadir un proyecto al control de código fuente, puede añadir archivos o grupos de archivos al sistema de control. Seleccione el archivo en la ventana Proyecto y haga clic en el comando **Proyecto | Control de código fuente | Agregar al control de código fuente**. Para seleccionar varios archivos mantenga pulsada la tecla **Ctrl** mientras hace clic en los archivos que desea añadir. Si ejecuta el comando en una carpeta de proyecto (de color verde), entonces se agregan al control de código fuente todos los archivos y subcarpetas de la carpeta.

© 2015 Altova GmbH



Cuando se agregan archivos al control de código fuente, la jerarquía de la carpeta local se copia al repositorio (no la jerarquía de la carpeta de proyecto). Es decir, si un archivo está en una subcarpeta de la carpeta local, entonces se crean en el repositorio la carpeta primaria y las demás carpetas antecesoras del archivo.

Cuando se añade el primer archivo de un proyecto al control de código fuente, se crean los enlaces correspondientes en el repositorio y se añade automáticamente el archivo de proyecto (archivo `.spp`). Para más información consulte el apartado Agregar al control de código fuente.

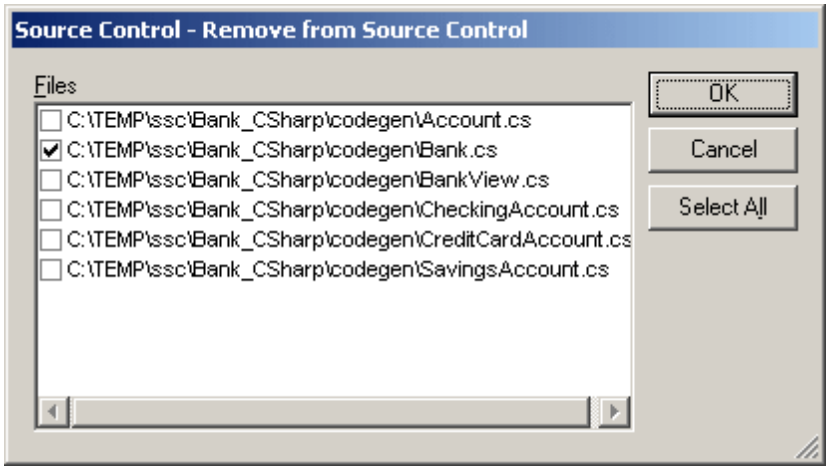
Símbolos de control de código fuente

Las carpetas y los archivos aparecen con un icono diferente dependiendo de su estado:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.
	Desprotegido localmente. Se puede editar y proteger.

Quitar del control de código fuente

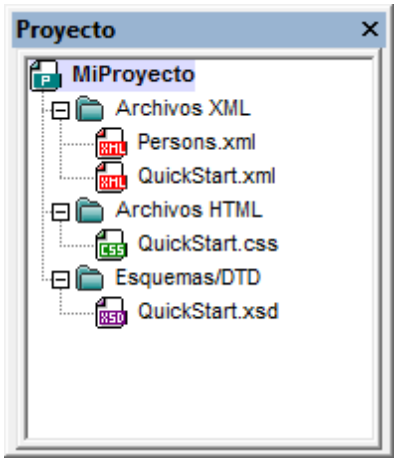
Para quitar un archivo del control de código fuente, selecciónelo y haga clic en el comando **Proyecto | Control de código fuente | Quitar del control de código fuente**. También puede quitar: (i) archivos de una carpeta de proyecto ejecutando el comando sobre la carpeta, (ii) varios archivos, si mantiene pulsada la tecla **Ctrl** mientras los selecciona y (iii) todo el proyecto, ejecutando el comando sobre el proyecto.



Compartir desde el control de código fuente

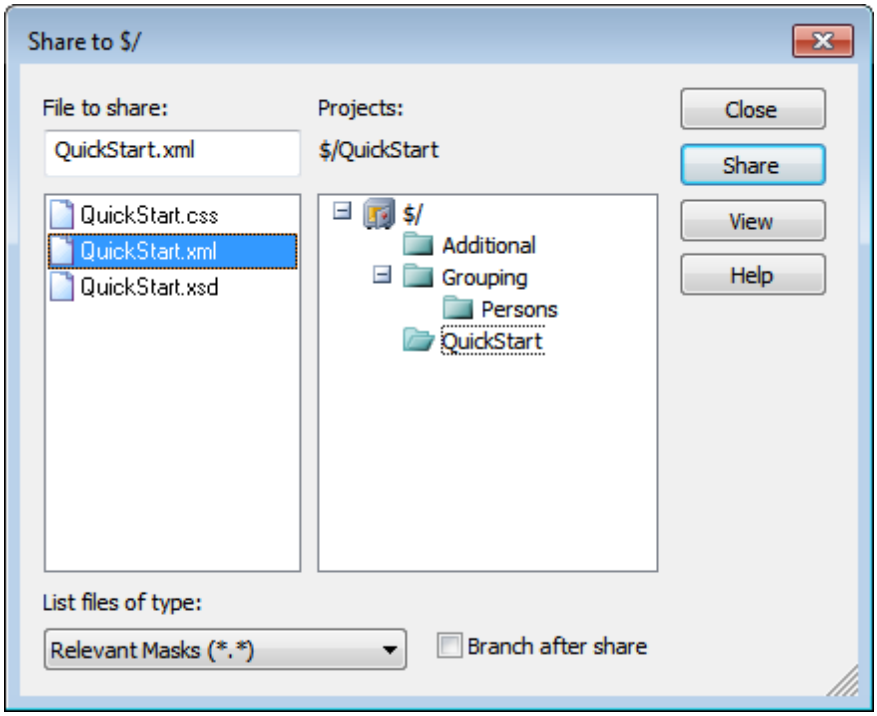
El comando **Compartir desde el control de código fuente** está disponible si su sistema de control de código fuente permite compartir contenido. Por ejemplo, puede compartir un archivo para que esté disponible en varias ubicaciones locales. Si se realizan cambios en uno de estos archivos locales, los cambios se reflejan en todas las versiones "compartidas" del archivo.

En la ventana Proyecto seleccione el proyecto (*imagen siguiente*) y después haga clic en **Compartir desde el control de código fuente**.



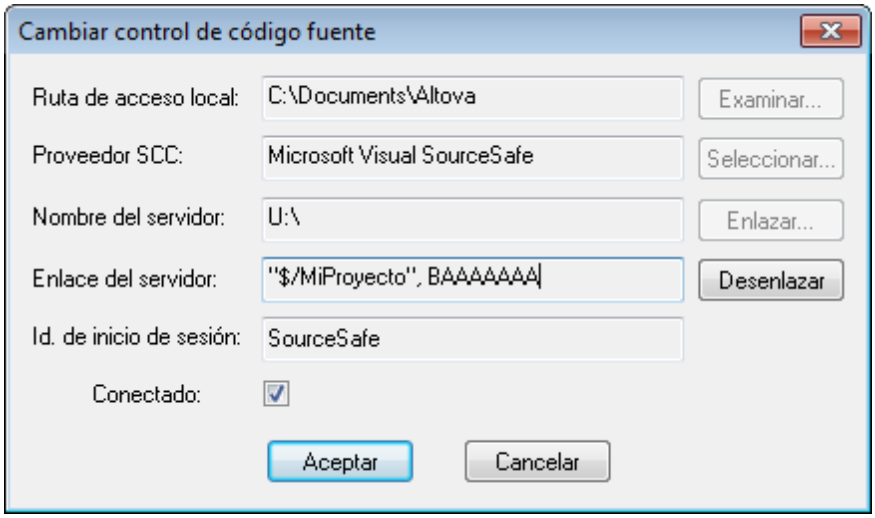
Aparece el cuadro de diálogo "Share To [carpeta]" (*imagen siguiente*).

© 2015 Altova GmbH



Ahora puede seleccionar los archivos que desea compartir. Primero seleccione en el panel derecho la carpeta donde están los archivos. Los archivos de la carpeta elegida aparecen en el panel izquierdo. Seleccione el archivo que desea compartir (para seleccionar varios pulse la tecla **Ctrl** mientras hace clic en los archivos). Los archivos seleccionados aparecen en el cuadro situado en la parte superior izquierda del diálogo. Haga clic en **Share** (*compartir*) y después en **Close** (cerrar) para copiar los archivos seleccionados a la carpeta local compartida.

El nombre de la carpeta compartida aparece en el nombre del cuadro de diálogo. En la imagen anterior, por ejemplo, se trata de la carpeta local (porque el signo `$` representa la carpeta del repositorio a la que está enlazada la carpeta local). En el cuadro de diálogo "Cambiar control de código fuente" puede ver y configurar la carpeta compartida (*imagen siguiente*) cambiando la ruta de acceso local y el enlace de servidor.

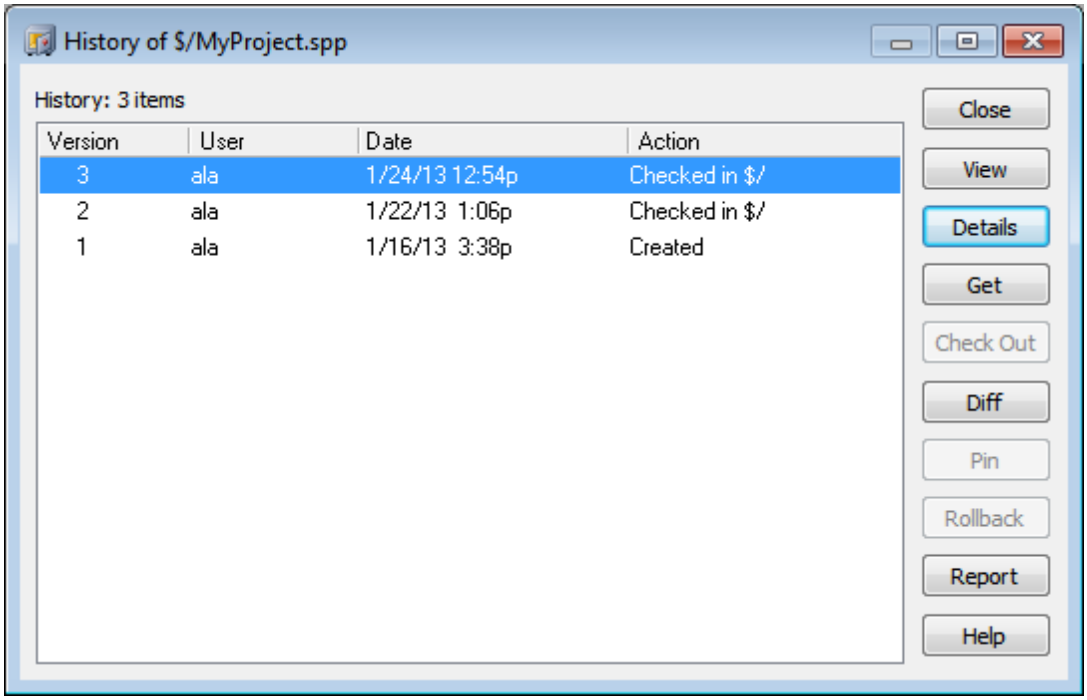


Para más información consulte la documentación de su sistema de control de código fuente.

Mostrar historial

El comando **Mostrar historial** activa la función del mismo nombre del sistema de control de código fuente activo. Muestra el historia del archivo seleccionado en la ventana Proyecto. Seleccione el título del proyecto para ver el historial del archivo de proyecto (archivo `.spp`). Puede ver información sobre las versiones previas de un archivo y las diferencias con la versión actual y recuperar versiones previas del archivo.

La siguiente imagen muestra el diálogo "Historial" del sistema de control de versiones Visual SourceSafe y enumera todas las versiones del archivo `MyProject.spp`.



Este cuadro de diálogo permite compartir y obtener versiones determinadas del archivo en cuestión. Al hacer doble clic en un archivo de la lista se abre el cuadro de diálogo "History details" (Historial detallado). Estos son los botones de este cuadro de diálogo:

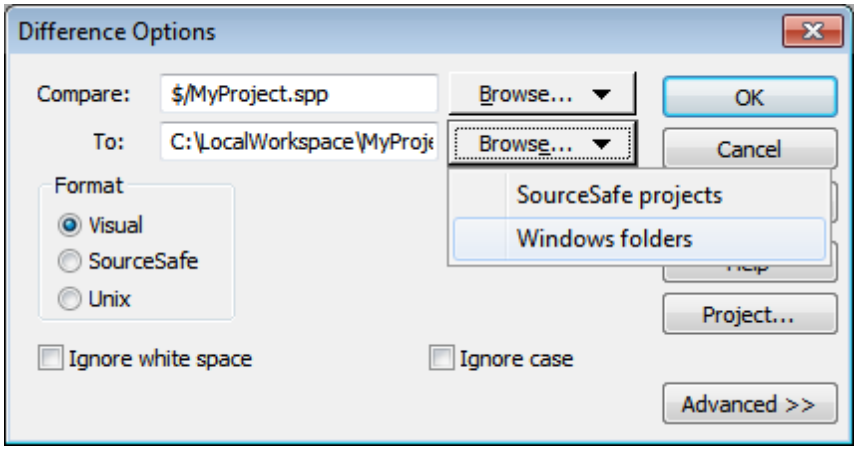
- ***Close:*** cierra el cuadro de diálogo.
- ***View:*** abre un cuadro de diálogo donde puede seleccionar el tipo de visor de archivos.
- ***Details:*** abre un cuadro de diálogo donde puede ver las propiedades del archivo activo actual.
- ***Get:*** recupera una de las versiones previas del archivo y la coloca en el directorio de trabajo.
- ***Check Out:*** desprotege una versión previa del archivo.
- ***Diff:*** abre el cuadro de diálogo Difference options, donde puede definir las opciones de comparación de las dos versiones del archivo. Marque dos versiones del archivo en esta ventana y después haga clic en **Diff** para ver las diferencias.
- ***Pin:*** ancla o desancla una versión del archivo, para que pueda definir la versión de archivo que desea usar al comparar dos archivos.
- ***Rollback:*** revierte el archivo a la versión seleccionada.
- ***Report:*** genera un informe de historial que se puede enviar a una impresora, a un

archivo o al portapapeles.
- ***Help:*** abre la ayuda en línea del complemento de control de código fuente.

Mostrar diferencias

El comando **Mostrar diferencias** está disponible si selecciona un archivo de la ventana Proyecto. Para seleccionar el archivo de proyecto (`.spp`) haga clic en el título del proyecto en la ventana Proyecto. El comando **Mostrar diferencias** inicia la herramienta de comparación del sistema de control de código fuente. Así puede ver las diferencias directamente en la aplicación de Altova.

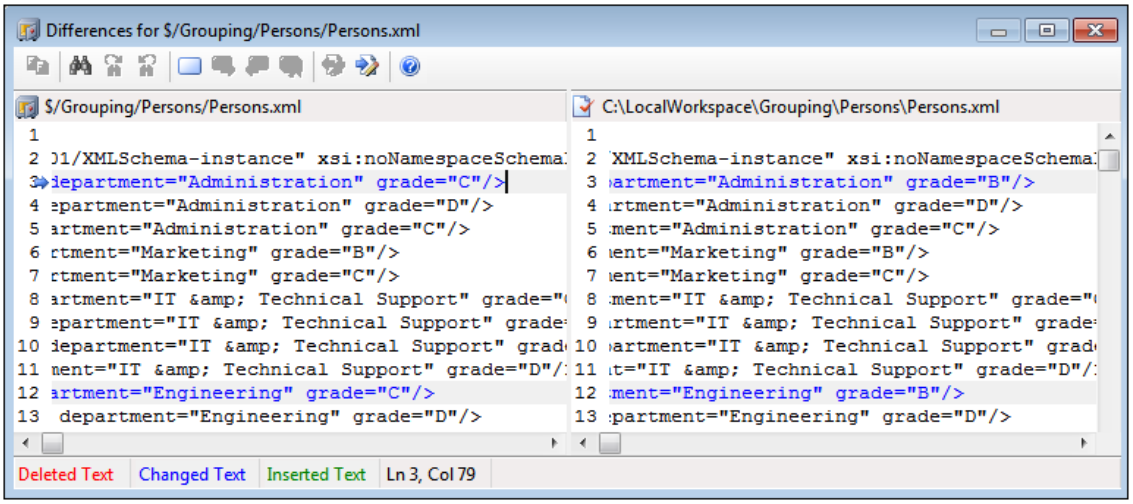
La imagen siguiente muestra la herramienta de comparación del sistema de control Visual SourceSafe.



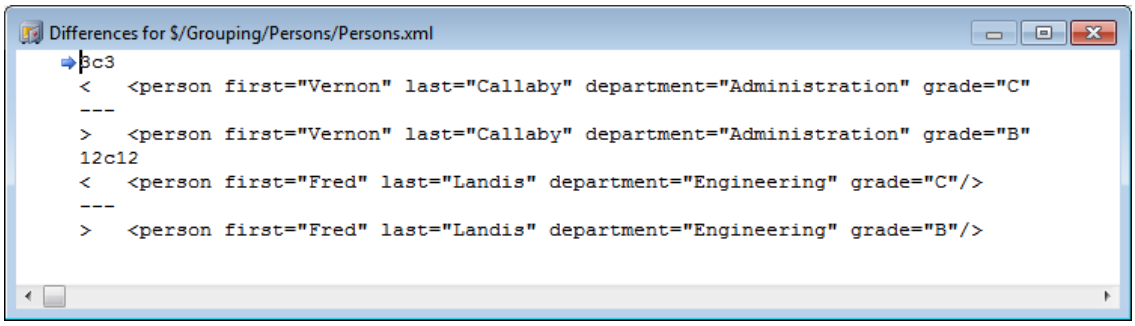
En los campos de texto *Compare* y *To* aparecen por defecto la versión del repositorio y la versión local respectivamente. Para buscar otros archivos:

1. Seleccione la opción *SourceSafe projects* en la lista desplegable del cuadro combinado **Examinar** (para buscar archivos del repositorio) o la opción *Windows folders* (para buscar carpetas locales).
2. Busque los archivos y selecciónelos.

Seleccione las opciones que desea usar y haga clic en **OK** para iniciar la comparación. Los resultados aparecen en otra ventana. Las imágenes que aparecen a continuación muestran los resultados de una misma comparación en dos formatos diferentes.



La primera imagen muestra los resultados de la comparación de Visual SourceSafe en formato visual (ver la opción *Format* > *Visual* del cuadro de diálogo de la primera imagen). La imagen que aparece a continuación muestra el resultado en formato Unix. En ambos casos, se encontraron dos diferencias.

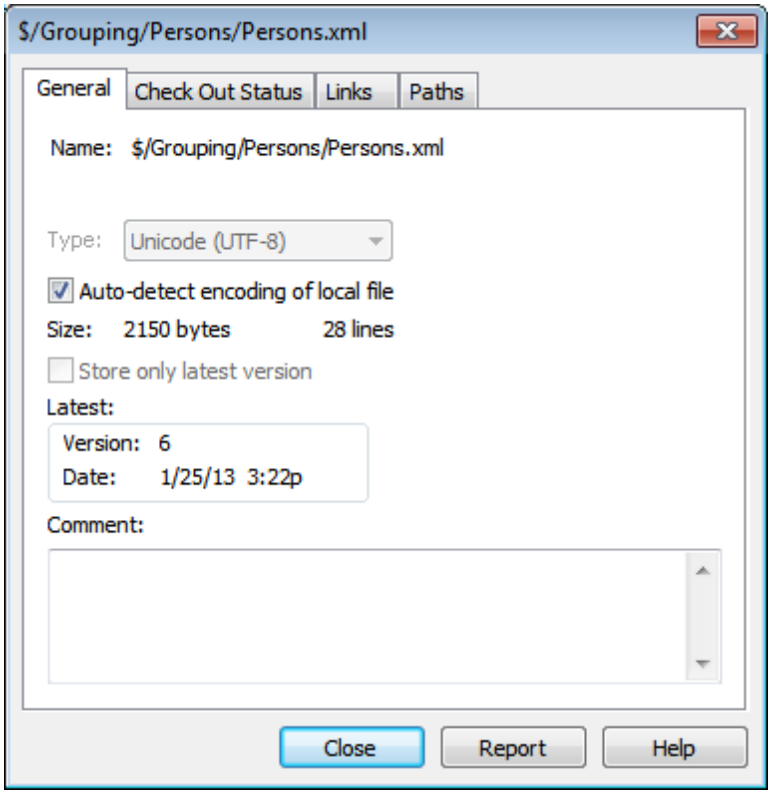


Para más información consulte la documentación de su sistema de control de código fuente.

Mostrar propiedades

El comando **Mostrar propiedades** muestra las propiedades del archivo que está seleccionado (*imagen siguiente*). Las propiedades que aparecen dependen del sistema de control de código fuente utilizado. La imagen siguiente corresponde al sistema Visual SourceSafe.

© 2015 Altova GmbH



Recuerde que este comando se habilita solamente si selecciona un solo archivo.

Para más información consulte la documentación de su sistema de control de código fuente.

Actualizar estado

El comando **Actualizar estado** actualiza el estado de todos los archivos del proyecto independientemente de cuál sea su estado actual.

Administrador del control de código fuente

El comando **Administrador del control de código fuente** inicia el software de control de código fuente en su interfaz nativa.

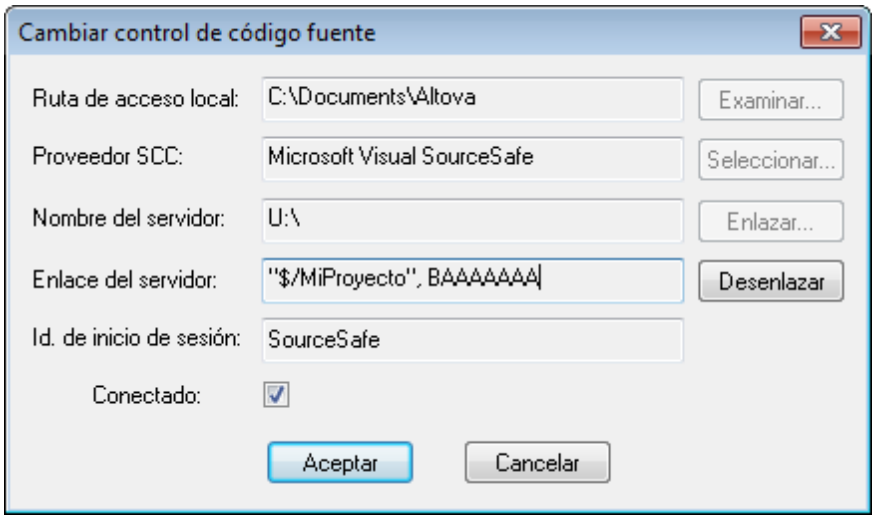
Cambiar control de código fuente

El enlace actual es lo que usa el proyecto de aplicación activo para conectarse a la BD del control de código fuente. Por tanto, en enlace actual debe ser correcto. Esto significa que el archivo de proyecto de la aplicación (archivo `.spp`) debe estar en la carpeta local y la carpeta enlazada del repositorio debe ser la base de datos donde se almacenan los archivos de este proyecto. Por lo general, la carpeta enlazada y sus subcarpetas se corresponden con la carpeta de la copia de trabajo local y sus subcarpetas.

En el cuadro de diálogo "Cambiar control de código fuente" (*imagen siguiente*) puede cambiar el

sistema de control (*Proveedor SCC*), la carpeta local (*Ruta de acceso local*) y el enlace al repositorio (*Nombre del servidor* y *Enlace del servidor*).

Las opciones solamente se pueden cambiar después de desenlazar el enlace actual. Para ello, pulse el botón **Desenlazar**. Ahora puede editar las demás opciones.



1. Con el botón **Examinar** puede buscar la carpeta local y con el botón Seleccionar puede seleccionar uno de los sistemas de control de código fuente instalados.
2. Después puede enlazar la carpeta local a una BD del repositorio. Para ello pulse el botón **Enlazar**. Aparece el cuadro de diálogo de conexión de sus sistema de control.
3. Si escribió un *Id. de inicio de sesión*, este identificador se pasa al sistema de control. De lo contrario, deberá escribir sus datos de inicio de sesión en el diálogo de conexión.
4. Seleccione la BD del repositorio a la que desea enlazar esta carpeta local.
5. Cuando termine haga clic en **Aceptar**.

© 2015 Altova GmbH

10.3.7 Agregar archivos al proyecto

El comando **Proyecto** | **Agregar archivos al proyecto** agrega archivos al proyecto actual. Use este comando para agregar archivos a cualquier carpeta del proyecto. Puede seleccionar un solo archivo o un grupo de archivos en el cuadro de diálogo Abrir. Si agrega archivos al proyecto (en lugar de agregarlos a una carpeta del proyecto), estos se distribuyen en las carpetas del proyecto dependiendo de las extensiones de tipo de archivo definidas en el cuadro de diálogo Propiedades del proyecto.

10.3.8 Agregar recurso global al proyecto

El comando **Proyecto** | **Agregar recurso global al proyecto** abre el cuadro de diálogo "Agregar recurso global", donde puede seleccionar un recurso global de tipo archivo o carpeta y agregarlo al proyecto. Si selecciona un recurso global de tipo archivo, el archivo se añade a la carpeta correspondiente según las extensiones de tipo de archivo definidas en el cuadro de diálogo Propiedades del proyecto. Si selecciona un recurso global de tipo carpeta, la carpeta se abre en un cuadro de diálogo "Abrir" y debe seleccionar un archivo. El archivo seleccionado en este cuadro de diálogo se añade a la carpeta de proyecto correspondiente según las extensiones de tipo de archivo definidas en el cuadro de diálogo Propiedades del proyecto. Para más información, consulte la sección del *Manual del usuario* dedicada a los recursos globales.

© 2015 Altova GmbH

10.3.9 Agregar URL al proyecto

El comando **Proyecto** | **Agregar URL al proyecto** agrega una URL al proyecto actual. Las direcciones URL de un proyecto incluyen su objeto de destino en el proyecto. Si ejecuta una operación por lotes en una URL o en una carpeta que contiene una URL, Authentic Desktop recupera el documento de la URL y ejecuta en él la operación solicitada.

10.3.10 Agregar archivo activo al proyecto

El comando **Proyecto** | **Agregar archivo activo al proyecto** añade el archivo activo al proyecto actual. Si acaba de abrir un archivo desde el disco duro o desde una URL, puede agregar el archivo al proyecto con este comando.

© 2015 Altova GmbH

10.3.11 Agregar archivo activo y relacionados al proyecto

El comando **Proyecto** | **Agregar archivo activo y relacionados al proyecto** añade al proyecto el archivo XML que está activo y todos los archivos relacionados con él. Si está trabajando con un documento XML que está basado en una DTD o en un esquema, este comando no sólo añade al proyecto el documento XML activo, sino también todos los archivos relacionados con él (por ejemplo, la DTD y todas las entidades externas analizadas a las que hace referencia la DTD).

Nota importante: no se consideran archivos relacionados los archivos a los que se hace referencia mediante instrucciones de procesamiento (por ejemplo, archivos XSLT).

10.3.12 Agregar carpeta de proyecto al proyecto

El comando **Proyecto** | **Agregar carpeta de proyecto al proyecto** añade una carpeta nueva al proyecto actual. Use este comando para agregar una carpeta nueva al proyecto actual o una subcarpeta a una carpeta de proyecto. Este comando también está disponible en el menú contextual que aparece al hacer clic con el botón secundario en una carpeta en la ventana Proyecto.

Nota: las carpetas de proyecto se pueden arrastrar y colocar en otra carpeta de proyecto o en otra posición del proyecto. Además, puede arrastrar carpetas desde el explorador de Windows hasta cualquier carpeta de la ventana Proyecto.

Nota: las carpetas de proyecto son verdes, mientras que las carpetas externas son amarillas.

© 2015 Altova GmbH

10.3.13 Agregar carpeta externa al proyecto

El comando **Proyecto** | **Agregar carpeta externa al proyecto** añade una carpeta externa nueva al proyecto actual. Use este comando para agregar una carpeta de red o local al proyecto actual. Este comando también está disponible en el menú contextual que aparece al hacer clic con el botón secundario en una carpeta en la ventana Proyecto.

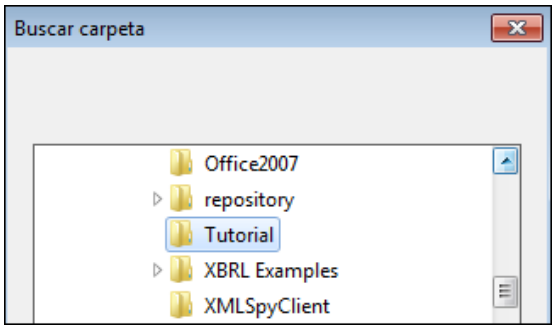
Nota: las carpetas externas son amarillas, mientras que las carpetas de proyecto son verdes.

Nota: los archivos de las carpetas externas no se pueden poner bajo control de código fuente.

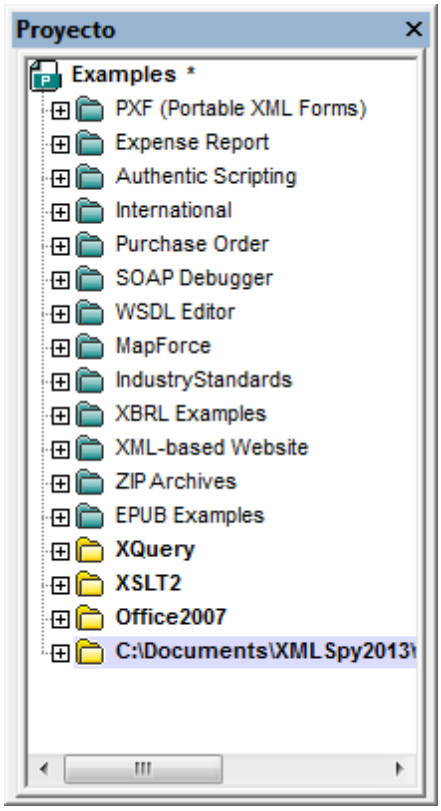
Agregar carpetas externas al proyecto

Siga estos pasos para agregar una carpeta externa al proyecto:

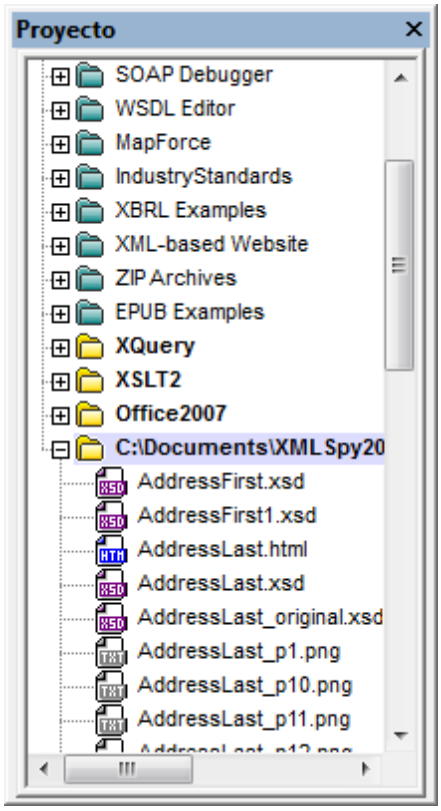
1. Seleccione la opción de menú **Proyecto | Agregar carpeta externa al proyecto**
2. En el cuadro de diálogo "Buscar carpeta" seleccione la carpeta que desea agregar y haga clic en **Aceptar**.



La carpeta seleccionada aparece en la ventana Proyecto.



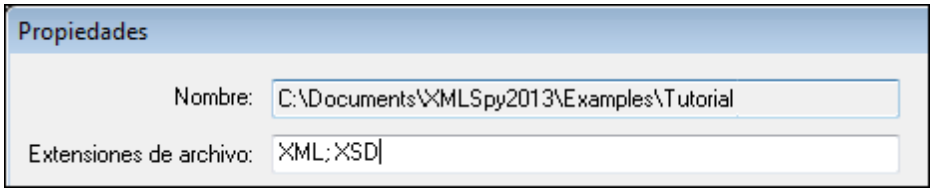
3. Haga clic en el icono + para ver el contenido de la carpeta.



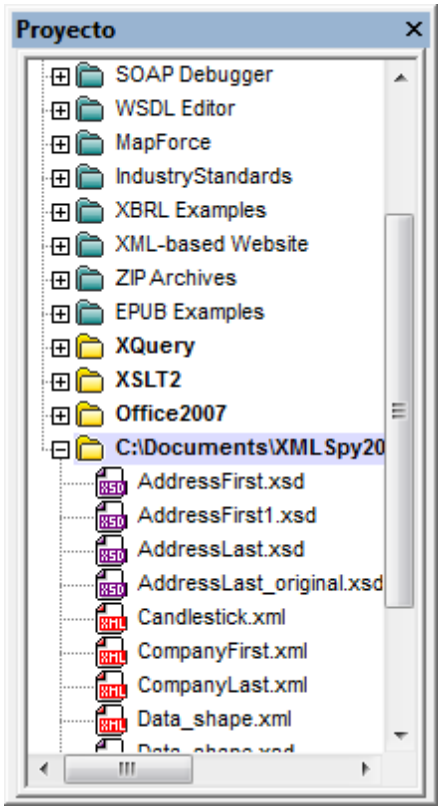
© 2015 Altova GmbH

Filtrar el contenido de las carpetas
Siga estos pasos para filtrar el contenido de la carpeta:

1. Haga clic con el botón secundario en la carpeta local y seleccione la opción de menú **Propiedades**. Aparece el cuadro de diálogo Propiedades.



2. En el campo *Extensiones de archivo:* escriba las extensiones de archivo de los tipos de archivo que desea ver. Para definir varios tipos de archivo, separe las extensiones con un **punto y coma** (*XML y XSD en el ejemplo de la imagen anterior*).
3. Para confirmar haga clic en **Aceptar**.



Ahora la ventana Proyecto muestra solamente los archivos XML y XSD de la carpeta externa seleccionada.

Validar carpetas externas
Siga estos pasos para validar y comprobar si los documentos de sus carpetas externas tienen un formato XML correcto:

1. Seleccione qué tipo de archivos desea ver y validar en la carpeta externa (siguiendo los pasos descritos más arriba).
2. Haga clic en la carpeta y después en el icono **Comprobar formato XML** o **Validar el documento XML** (**F7** y **F8** respectivamente). A continuación se comprueban y

validan todos los archivos visibles en esa carpeta. Si un archivo tiene un formato XML incorrecto o no es válido, el archivo se abre en la ventana principal.

3. Corrija el error y ejecute de nuevo el proceso de validación para asegurarse de que se solucionó el problema.

Actualizar las carpetas de proyecto externas

Es posible que añada o elimine archivos en la carpeta local o de red. Cuando lo haga, será necesario actualizar la vista de la carpeta externa. Para ello, haga clic con el botón secundario en la carpeta externa y seleccione la opción **Actualizar** en el menú contextual.

Eliminar carpetas externas y sus archivos

Para eliminar la carpeta externa de la ventana Proyecto, selecciónela y pulse la tecla **Suprimir**. También puede hacer clic con el botón secundario en la carpeta externa y seleccionar el comando **Eliminar**. Recuerde que la carpeta sólo se elimina en la ventana Proyecto, pero no se elimina ni en el disco duro ni en la red.

Para eliminar un archivo de una carpeta externa es necesario eliminar el archivo en el disco duro o en la red. Para que el cambio se refleje en el proyecto es necesario actualizar el contenido de la carpeta (haga clic en la carpeta con el botón secundario y seleccione **Actualizar**).

Nota: las carpetas externas se pueden arrastrar y colocar en otra carpeta de proyecto o en otra posición del proyecto. Además, puede arrastrar carpetas externas desde el explorador de Windows hasta cualquier carpeta de la ventana Proyecto.

© 2015 Altova GmbH

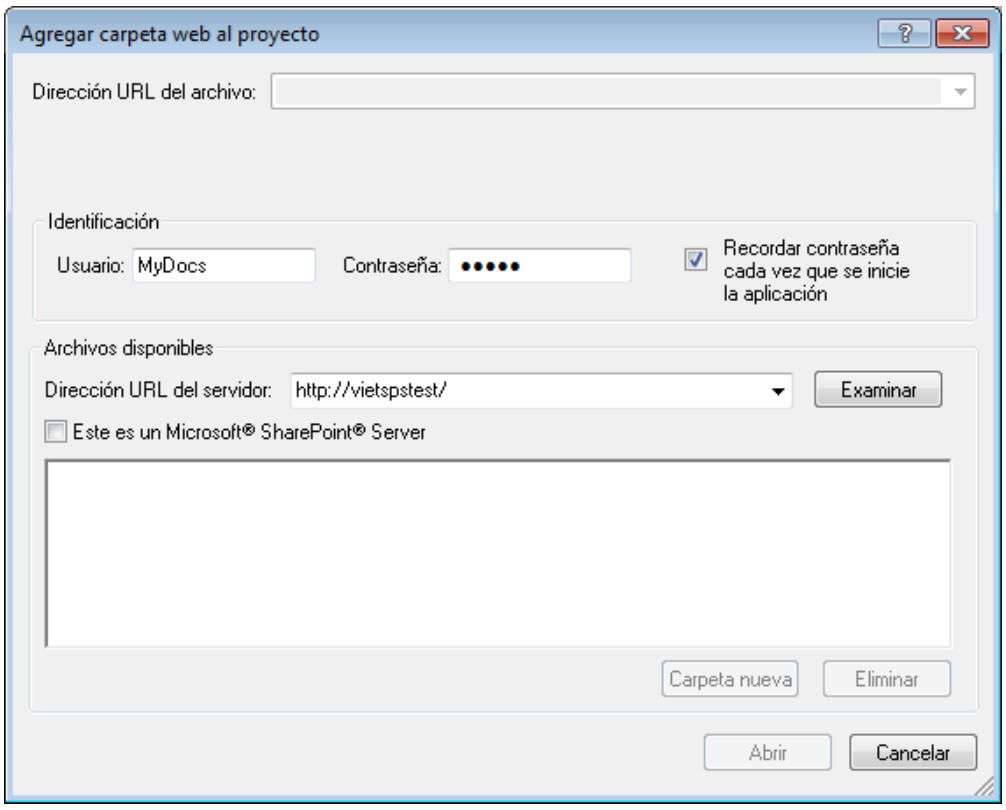
10.3.14 Agregar carpeta web externa al proyecto

Este comando añade una carpeta web externa al proyecto actual. Este comando también está disponible en el menú contextual que aparece al hacer clic con el botón secundario en una carpeta de la ventana Proyecto. Recuerde que los archivos de carpetas externas no se pueden poner bajo control de código fuente.

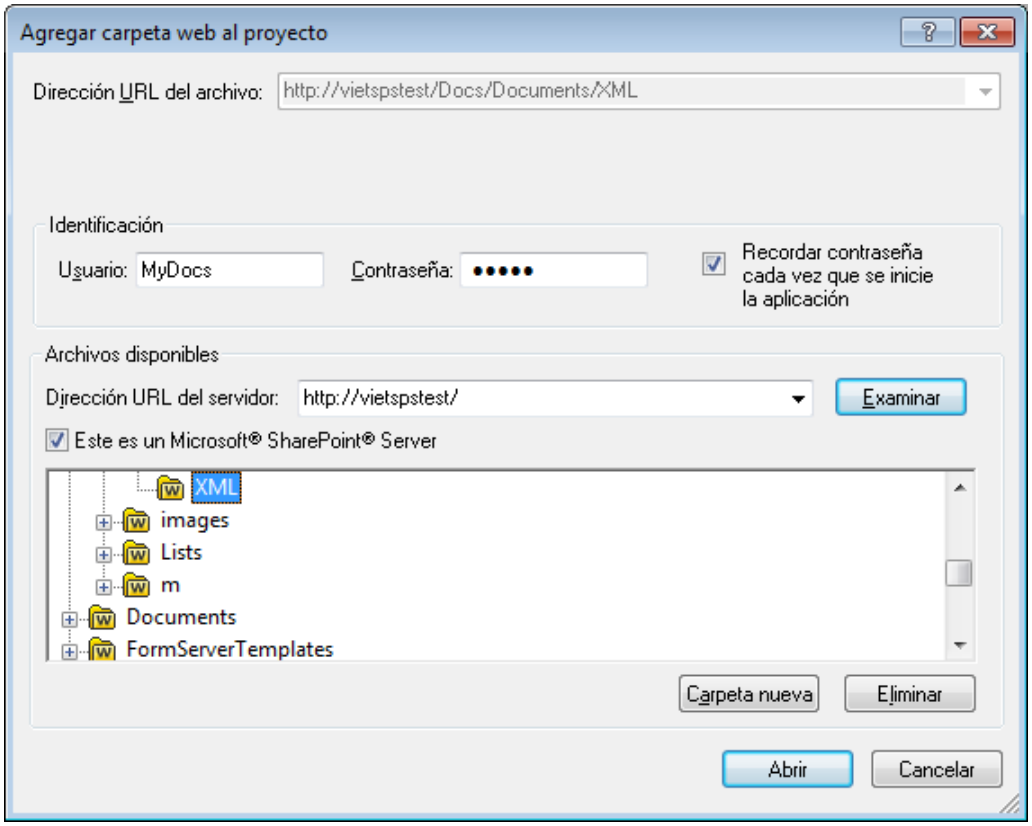
Agregar carpetas web externas al proyecto

Siga estos pasos para agregar una carpeta web externa al proyecto:

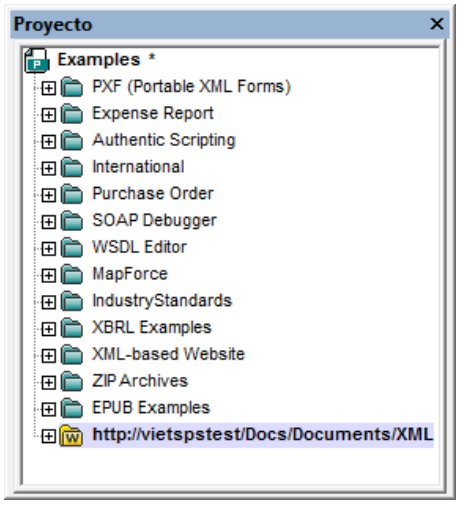
1. Seleccione la opción de menú **Proyecto | Agregar carpeta web externa al proyecto**. Aparece el cuadro de diálogo "Agregar carpeta web al proyecto" (*imagen siguiente*).



2. Haga clic en el campo *Dirección URL del servidor* y escriba la URL del servidor. Si se trata de un servidor Microsoft® SharePoint® Server, marque la casilla. Consulte el subapartado titulado ***Carpetas de los servidores Microsoft® SharePoint® Server*** de esta página para más información sobre cómo trabajar con archivos de este tipo de servidor.
3. Si el servidor está protegido con contraseña, escriba su Id. de usuario y contraseña en los campos correspondientes.
4. Haga clic en **Examinar** para conectarse al servidor y ver las carpetas que están disponibles.

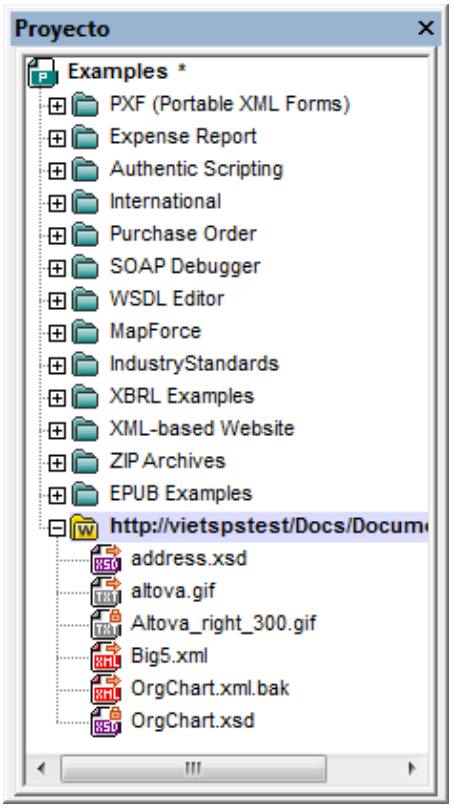


5. Haga clic en la carpeta que desea agregar al proyecto. El botón **Abrir** no se habilita hasta que no seleccione una carpeta. La URL de la carpeta aparece ahora en el campo *Dirección URL del archivo*.
6. Haga clic en **Abrir** para agregar la carpeta al proyecto.



7. Haga clic en el icono + para ver el contenido de la carpeta.

 © 2015 Altova GmbH



Filtrar el contenido de las carpetas

Para filtrar el contenido de una carpeta, haga clic con el botón secundario en la carpeta y seleccione **Propiedades** en el menú contextual. Aparece el cuadro de diálogo Propiedades. Haga clic en el campo *Extensiones de archivo* y escriba los tipos de archivo que desea ver en la carpeta (por ejemplo, archivos XML y XSD). Separe cada tipo de archivo con un punto y coma (por ejemplo, `xml; xsd; sps`). A partir de ahora la ventana Proyecto muestra solamente los tipos de archivo que tienen las extensiones indicadas.

Validar carpetas web externas

Para validar los archivos de una carpeta y comprobar si tienen un formato XML correcto, seleccione la carpeta y después haga clic en el icono **Comprobar formato XML** o **Validar el documento XML** (**F7** y **F8** respectivamente). La operación se ejecuta en todos los archivos que están visibles en la carpeta. Si un archivo no es válido o tiene un formato XML incorrecto, entonces el archivo se abre en la ventana principal para que pueda editarlo. Corrija el error y reinicie el proceso para volver a revisar el resto de la carpeta. Recuerde que también puede seleccionar varios archivos de la carpeta si mantiene pulsada la tecla **Ctrl** mientras hace clic en los archivos uno por uno. En este caso, al pulsar **F7** o **F8**, se revisan únicamente los archivos seleccionados.

Actualizar el contenido de las carpetas de proyecto

Es posible que se añadan o eliminen archivos en la carpeta web. Cuando esto ocurra, será necesario actualizar la vista de la carpeta externa. Para ello, haga clic con el botón secundario en la carpeta externa y seleccione la opción **Actualizar** en el menú contextual.

Eliminar carpetas y archivos

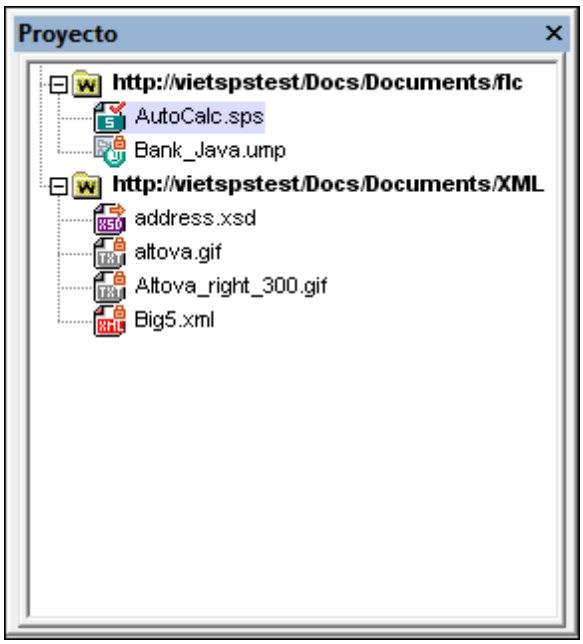
Puesto que lo que se añadió al proyecto es la carpeta web, solamente se puede eliminar del proyecto la carpeta web y no los archivos que están dentro de la carpeta. Para eliminar una carpeta web del proyecto, tiene dos opciones: (i) puede hacer clic con el botón secundario en la carpeta y seleccionar **Eliminar** o (ii) puede seleccionar la carpeta y pulsar la tecla **Suprimir**. Recuerde que la carpeta se elimina solamente en la ventana Proyecto, no en el servidor web.

Nota: si en la ventana Proyecto hace clic con el botón secundario en un archivo de la carpeta web y pulsa la tecla **Suprimir**, el archivo no desaparece de la ventana Proyecto. Para que desaparezca es necesario eliminar el archivo en el servidor y después actualizar el contenido de la carpeta externa.

Carpetas de los servidores Microsoft® SharePoint® Server

Cuando añada al proyecto una carpeta de un servidor Microsoft® SharePoint® Server, los archivos de la carpeta se pueden proteger y desproteger con los comandos del menú contextual de la lista de archivos de la ventana Proyecto (*imagen siguiente*). Para usar estos comandos haga clic con el botón secundario en el archivo con el que desea trabajar y seleccione el comando pertinente (**Desproteger**, **Proteger** o **Deshacer desprotección**).

El Id. de usuario y la contraseña se pueden guardar en las propiedades de las carpetas del proyecto para que no sea necesario pasar por el proceso de verificación cada vez que acceda al servidor.



En la ventana Proyecto (*imagen anterior*), los iconos de archivo tienen símbolos que indican el estado de los archivos en el servidor:

	Protegido. Se puede desproteger.
	Desprotegido por otro usuario. No se puede desproteger.

© 2015 Altova GmbH

	Desprotegido localmente. Se puede editar y proteger.

Recuerde que:

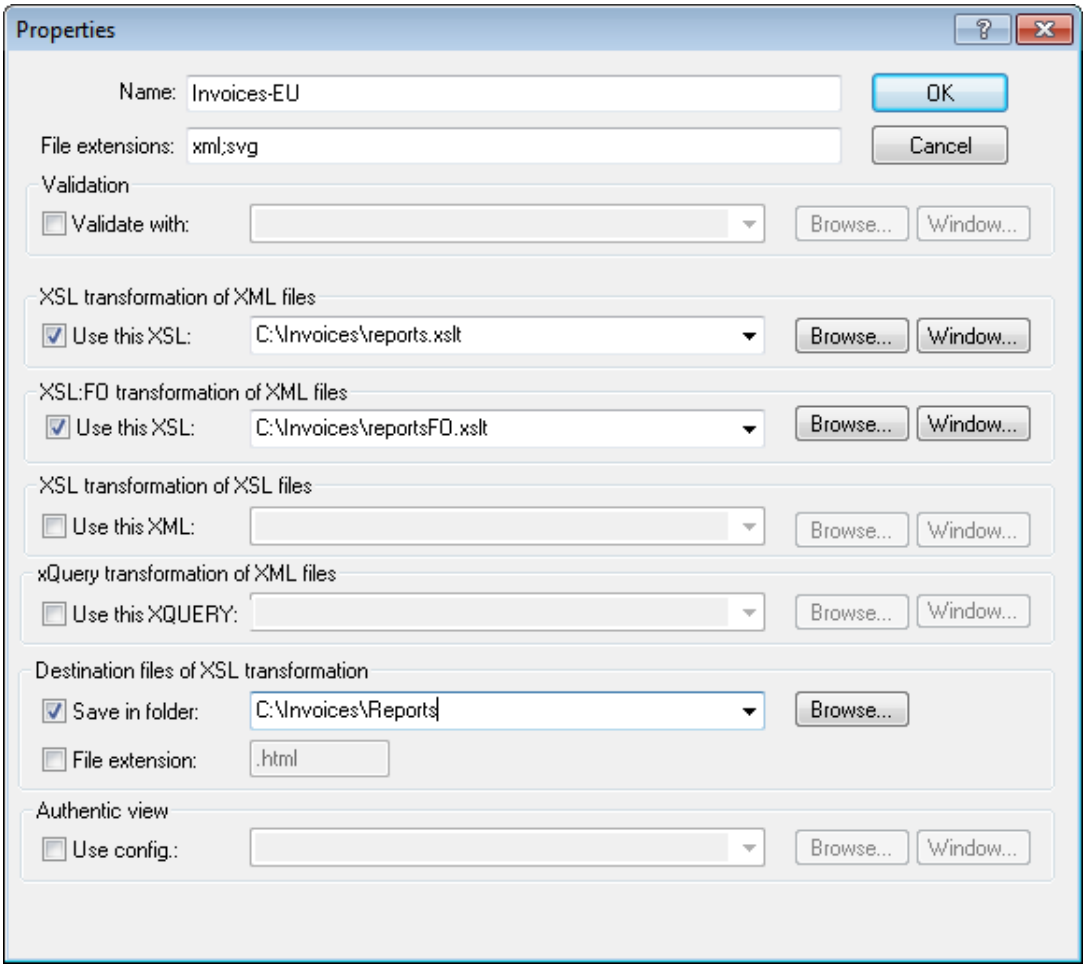
- Después de desproteger un archivo, puede editarlo en su aplicación de Altova y guardarlo con el comando **Archivo | Guardar (Ctrl+S)**.
- Puede proteger el archivo editado con el menú contextual de la ventana Proyecto (*imagen anterior*) o con el menú contextual que aparece al hacer clic con el botón secundario en la pestaña de archivo de la ventana principal de la aplicación (*imagen siguiente*).

- Cuando otro usuario desprotege un archivo, este archivo no se puede desproteger.
- Cuando desproteja un archivo localmente, puede deshacer la desprotección con el comando **Deshacer desprotección** del menú contextual. Como resultado se devuelve el archivo al servidor sin ningún cambio.
- Si desprotege un archivo en una aplicación de Altova, no lo puede desproteger en otra aplicación de Altova porque se entiende que está desprotegido. A partir de ese momento los comandos disponibles en las aplicaciones de Altova compatibles con Microsoft® SharePoint® Server serían **Proteger** y **Deshacer desprotección**.

10.3.15 Propiedades

El comando **Proyecto** | **Propiedades** abre el cuadro de diálogo *Propiedades (captura de pantalla siguiente)* del proyecto activo. Si hace clic con el botón secundario sobre una carpeta de la ventana Proyecto (y no en la misma carpeta del proyecto) y selecciona **Propiedades**, se abrirá el cuadro de diálogo Propiedades de esa carpeta. La configuración del cuadro de diálogo se describe a continuación.

Nota: Si el archivo de su proyecto aparece en el Control de código fuente, aparece un mensaje que le pregunta si desea desproteger el archivo del proyecto (`.spp`). Haga clic en **Aceptar** si desea editar la configuración y poder guardarla.



Configuración

Extensiones de archivo
La configuración *Extensiones de archivo* se activa para carpetas individuales y no para la carpeta del proyecto. Cuando se agrega un archivo al proyecto, se agrega a la carpeta en la que se ha definido la extensión de su archivo. Por ejemplo, supongamos que agregamos un archivo al proyecto llamado `MyReport.xml`. Si las extensiones del archivo `.xml` se han configurado en

carpeta `Invoices-EU` (tal y como se muestra en la captura de pantalla anterior) entonces se añadirá el archivo `MyReport.xml` a la carpeta `Invoices-EU`. Si existe más de una carpeta a la que desee agregar los archivos XML, entonces deberá agregar directamente a la carpeta archivos XML individuales (en vez de al proyecto).

Id. de usuario y contraseña para carpetas externas
En las carpetas externas (incluidas las carpetas web externas) puede guardar el Id. de usuario y la contraseña necesarios para acceder al servidor.

Validación
El DTD, el esquema XML o esquema JSON que se deben usar para validar los archivos en la carpeta actual (o el proyecto completo si las propiedades son las del proyecto).

Transformación XSL de archivos XML
La hoja de estilos XSLT se usa par la transformación XSLT de archvivos XML en la carpeta.

Transformación XSL:FO de archivos XML
La hoja de estilos XSLT para transformar archivos XML a XSL-FO en la carpeta.

Transformación XSL de archivos XSL
El archivo XML que se va ausar para la transformación XSLT con archivos XSLT en la carpeta.

Transformación XQuery de archivos XML
El archivo XQuery que se va a usar para las ejecuciones XQuery de archivos XML en la carpeta.

Archivos de destino de la transformación XSL
El directorio de destino de las transformaciones XSL y de forma opcional, la extensión del archivo del documento resultante.

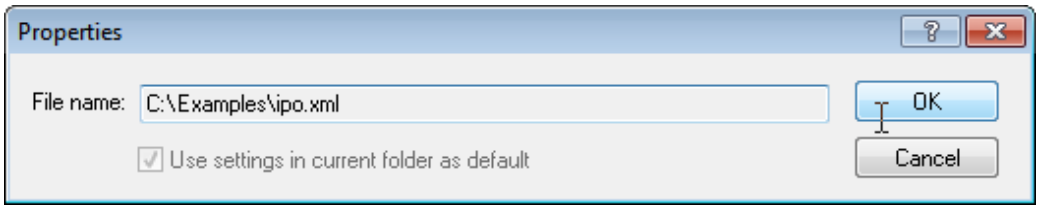
Vista authentic
La opción *Utilizar configuración* permite seleccionar un archivo SPS (StyleVision Power Stylesheet) para la edición de archivos XML en la vista Authentic. Observe que el archivo XML debe de validarse de acuerdo al mismo esquema que se usó para el SPS.

Notas sobre las propiedades del proyecto

Observaciones relativas a la preferencia o prioridad
Tenga en cuenta lo siguiente:

- Cuando lleve a cabo las validaciones o transformaciones XSLT/XQuery a través los menús contextuales de la carpeta del proyecto, entonces los archivos de validación o transformación especificados en este cuadro de diálogo tienen preferencia sobre cualquier otra asignación del archivo XML. Además, la configuración especificada para carpetas de proyectos individuales tiene preferencia sobre la configuración especificada para carpetas anteriores.
- Si un archivo se encuentra presente en varias carpetas del proyecto a se le ha asignado diferentes archivos de validación o transformación en carpetas diferentes, entonces puede configurar qué asignación usar cuando el archivo se procese fuera del proyecto. Especifíquelo de la siguiente forma: Ubique el archivo en la carpeta del proyecto cuyas

asignaciones desea utilizar. Haga clic con el botón secundario del ratón sobre el archivo que se encuentra en la carpeta del proyecto y seleccione **Propiedades**. En el cuadro de diálogo que aparece (*captura de pantalla siguiente*), seleccione *Usar la configuración predeterminada en la carpeta actual* (La carpeta actual es la carpeta del proyecto en la que se encuentra ubicada el archivo). Si se deshabilita esta opción, significa que la configuración de la carpeta actual ya se encuentra seleccionada como la configuración por defecto que se va a utilizar. Si selecciona un archivo de instancia que se encuentre en la carpeta del proyecto y que no viene predeterminada por defecto, entonces la opción se activa y puede cambiar la configuración predeterminada para que se convierta en la configuración de esta carpeta. Observe que si el archivo presenta una asignación local (es decir, una asignación dentro del archivo en sí mismo), entonces se usará la asignación local y se ignorará la configuración por defecto de la carpeta.



 © 2015 Altova GmbH

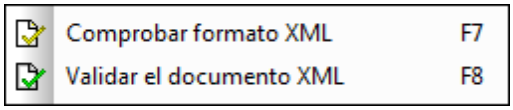
10.3.16 Proyectos usados recientemente

Este comando muestra el nombre de archivo y la ruta de acceso de los últimos nueve proyectos utilizados, para que pueda abrirlos rápidamente.

Recuerde que Authentic Desktop abre automáticamente el último proyecto utilizado nada más iniciarse la aplicación (**Herramientas | Opciones | pestaña *Archivo,*** *Proyecto* | *Abrir el último proyecto al iniciarse el programa*).

10.4 Menú XML

Los comandos del menú **XML** sirven para trabajar con documentos XML.



Entre las tareas XML más comunes se encuentran la comprobación de formato XML y la validación de documentos XML. Los comandos para estas operaciones se encuentran en este menú.

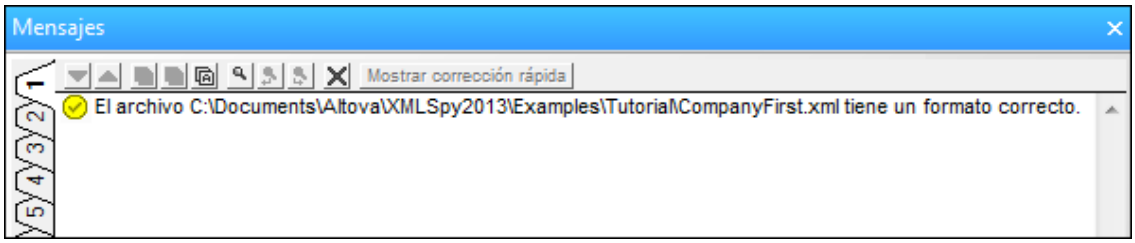
© 2015 Altova GmbH

10.4.1 Comprobar formato XML

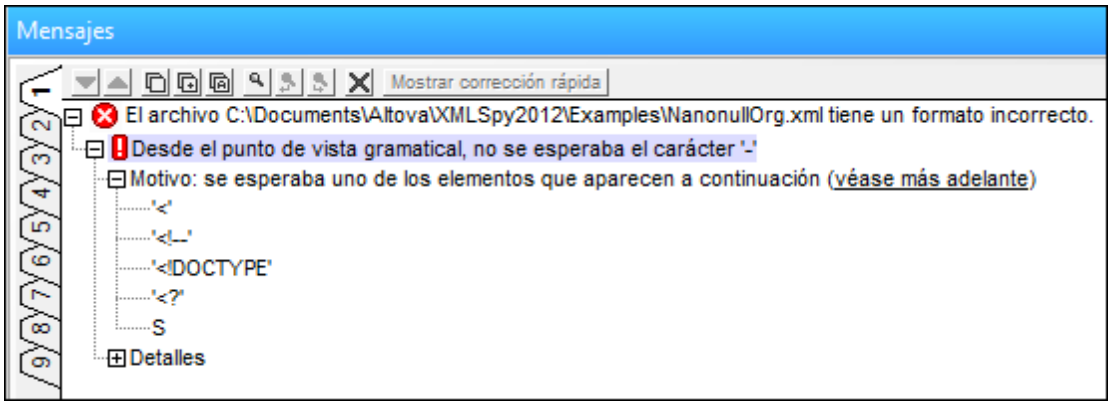
F7

El comando **XML | Comprobar formato XML** (**F7**) revisa el documento activo y comprueba si el formato es correcto según las definiciones de la especificación XML 1.0. Es obligatorio que el documento XML tenga un formato XML correcto. Authentic Desktop hace una comprobación de formato XML cada vez que se abre o guarda un documento.

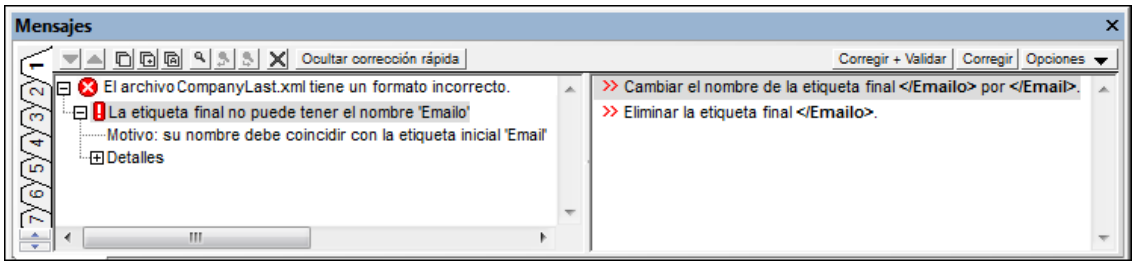
Si el formato XML del documento es correcto, aparece un mensaje parecido a este en la ventana Mensajes:



Si se detecta un error durante la comprobación de formato XML, en la ventana Mensajes aparece un mensaje de error explicando el problema:



A partir de la información disponible en el esquema, XMLSpy le sugiere opciones de corrección rápida si la comprobación de formato XML se realizó en la **vista Texto o Cuadrícula**. Para ver una lista de opciones de corrección rápida, pulse el botón **Mostrar corrección rápida** (*imagen anterior*). En la ventana Mensajes aparece un panel con las opciones de corrección rápida (*imagen siguiente*). Recuerde que la ventana Mensajes muestra los errores uno por uno.


Mensajes
Ocultar corrección rápida
Corregir + Validar
Corregir
Opciones
El archivo CompanyLast.xml tiene un formato incorrecto.
La etiqueta final no puede tener el nombre 'Emailo'
Motivo: su nombre debe coincidir con la etiqueta inicial 'Email'
Detalles
Cambiar el nombre de la etiqueta final </Emailo> por </Email>.
Eliminar la etiqueta final </Emailo>.

En el panel de corrección rápida seleccione una de las sugerencias y haga clic en el botón **Corregir + Validar** o en el botón **Corregir** (*imagen anterior*). El error se corregirá con la opción seleccionada. Si lo prefiere, también puede hacer doble clic en la opción que más le convenga. Al

hacer doble clic en la opción, pueden pasar dos cosas, dependiendo de las opciones elegidas en la lista desplegable *Opciones*: (i) se corrige el error o (ii) se corrige el error y se valida el documento. El comando **Corregir + Validar** es muy práctico porque el validador revisa todo el documento y no sólo detecta errores de validación sino también de formato XML.

Para ocultar el panel de corrección rápida, pulse el botón **Ocultar corrección rápida** (*imagen anterior*).

Nota: la ventana Mensajes tiene nueve pestañas. El resultado de la validación siempre aparece en la pestaña activa. Es decir, puede validar un documento XML en la pestaña 1, pasar a la pestaña 2 y validar otro documento XML y así sucesivamente. De este modo puede comparar los resultados obtenidos y guardados en cada pestaña. Si no cambia de pestaña y valida otro documento, los resultados de la pestaña 1 se sobrescriben con los resultados de la nueva validación.

Validación desde la ventana Proyecto

El comando **Validar el documento XML** también se puede aplicar a un archivo, carpeta o grupo de archivos del proyecto activo. Seleccione el archivo o carpeta en la ventana Proyectos y después haga clic en **XML | Validar el documento XML (F8)**. Los archivos no válidos del proyecto se abren en la ventana principal y aparece el mensaje de error "El archivo no es válido".

Nota: la ventana Mensajes tiene nueve pestañas. El resultado de la comprobación de formato siempre aparece en la pestaña activa. Es decir, puede comprobar el formato XML de un documento en la pestaña 1, pasar a la pestaña 2 y comprobar el formato de otro documento XML y así sucesivamente. De este modo puede comparar los resultados obtenidos y guardados en cada pestaña. Si no cambia de pestaña y comprueba el formato de otro documento, los resultados de la pestaña 1 se sobrescribirán con los resultados de la nueva comprobación.

Por lo general, no está permitido guardar un documento XML cuyo formato es incorrecto, pero Authentic Desktop le permite guardar el documento de todos modos. Esta opción es muy práctica si desea posponer el trabajo y terminar el documento más adelante.

Nota: también puede usar el comando **Comprobar formato XML** con cualquier archivo, carpeta o grupo de archivos de la ventana Proyecto. Para ello haga clic en el archivo, la carpeta o el grupo de archivos y haga clic en el icono **Comprobar formato XML**.

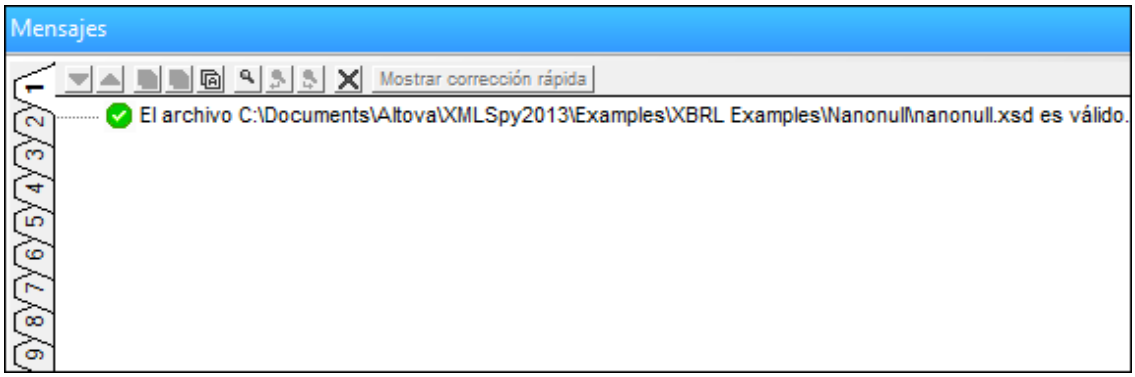
© 2015 Altova GmbH

10.4.2 Validar el documento XML

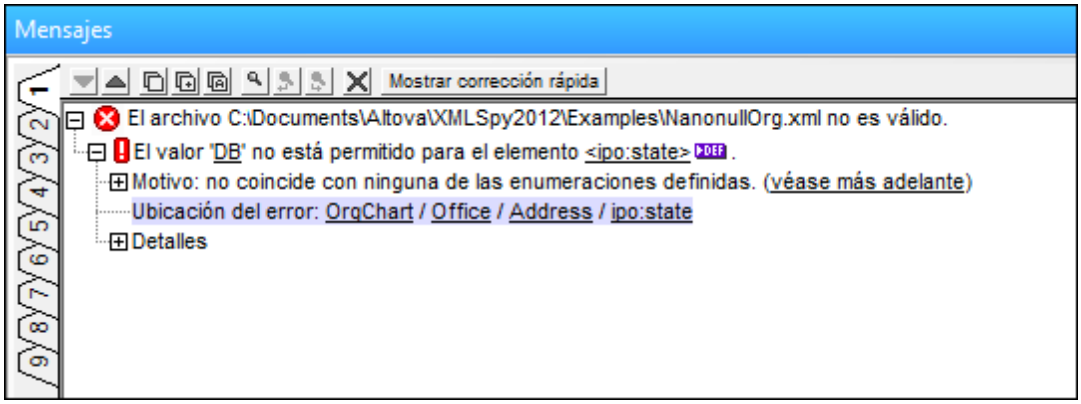
F8

El comando **XML | Validar el documento XML** (**F8**) sirve para validar documentos XML con esquemas XML, DTD y otros esquemas. Si lo desea, puede configurar la aplicación para que valide automáticamente los documentos al abrirlos o guardarlos (**Herramientas | Opciones | Archivo**). El comando **Validar el documento XML** también realiza una comprobación de formato XML antes de validar el documento. Es decir, si va a validar el documento, no hace falta usar el comando **Comprobar formato XML** antes del comando **Validar el documento XML**.

Si el documento es válido, aparece este mensaje en la ventana Mensajes:



Si el documento no es válido, entonces aparece un mensaje describiendo el error (*imagen siguiente*). Para ir al nodo del documento XML que contiene el error, haga clic en el enlace que incluye el mensaje.


Mensajes
Mostrar corrección rápida
El archivo C:\Documents\Altova\XMLSpy2012\Examples\NanonullOrg.xml no es válido.
El valor 'DB' no está permitido para el elemento <ipo:state>.
Motivo: no coincide con ninguna de las enumeraciones definidas. (véase más adelante)
Ubicación del error: OrgChart / Office / Address / ipo:state
Detalles

Nota: la ventana Mensajes tiene nueve pestañas. El resultado de la validación siempre aparece en la pestaña activa. Es decir, puede validar el documento XML en la pestaña 1, pasar a la pestaña 2 y validar otro documento XML y así sucesivamente. De este modo puede comparar los resultados obtenidos y guardados en cada pestaña. Si no cambia de pestaña y valida otro documento, los resultados de la pestaña 1 se sobrescribirán con los resultados de la nueva validación.

Validación desde la ventana Proyecto

El comando **Validar el documento XML** también se puede aplicar a un archivo, una carpeta o un grupo de archivos del proyecto activo. En la ventana Proyecto seleccione el archivo, la carpeta

o el grupo de archivos que desea validar. Después haga clic en **XML | Validar el documento XML** o pulse **F8**. Los archivos que no sean válidos se abren en la ventana principal y aparece un mensaje de tipo *El archivo no es válido* en la ventana Mensajes.

Automatizar la validación con RaptorXML 2016

RaptorXML es una aplicación independiente de Altova para validar XML, transformar XSLT y ejecutar XQuery. Se puede usar desde la línea de comandos, mediante la interfaz de COM, en programas Java y en aplicaciones .NET, por tanto, puede automatizar sus tareas de validación con ayuda de RaptorXML.

Por ejemplo, puede crear un archivo batch que llame a RaptorXML para ejecutar la validación en un conjunto de documentos y que envíe el resultado de la validación a un archivo de texto. Para más información, consulte la documentación de RaptorXML en nuestro sitio web.

© 2015 Altova GmbH

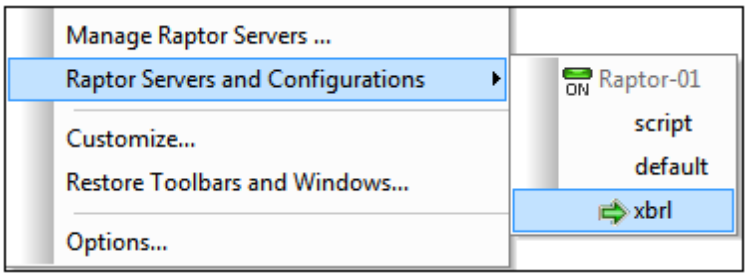
10.4.3 Validar el documento XML en el servidor

Ctrl+F8

El comando **XML** | **Validar en el servidor (alto rendimiento) (Ctrl+F8)** valida el documento XML activo en el servidor XML activo actualmente usando la configuración activa. El comando lleva a cabo la validación y muestra los resultados en la ventana "Mensajes".

Note: El rendimiento actual depende del número de los núcleos del procesador del equipo usados por el RaptorXML Server en la validación: Cuanto más elevado sea el número de núcleos utilizados, más rápido será el procesamiento.

Si ha definido varias configuraciones o varios servidores, puede seleccionar un servidor o uno de sus configuraciones como la configuración activa. La configuración activa se usará para las validaciones siguientes. Al colocar el cursor sobre el comando **Herramientas | Raptor Servers y Configuraciones** *(consulte la captura de pantalla siguiente),* aparece un submenú que contiene todos los servidores que se han agregado junto con las configuraciones de cada uno. Seleccione la configuración del servidor que desee activar. En la imagen siguiente puede observar que se ha seleccionado la configuración `xbrl` del servidor `Raptor-01` como configuración activa (tal y como lo indica la flecha verde).



Observación: Puede seleccionar la configuración activa en el menú desplegable del icono **Validar en el servidor** .. Este menú también presenta un comando para validar el EDGAR en el servidor activo.

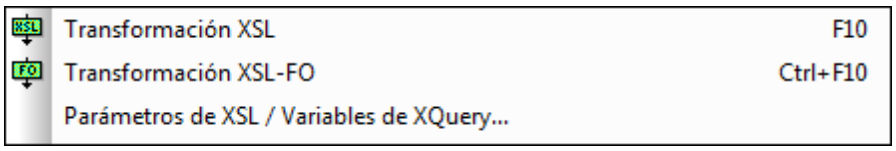
El comando **Validar documento XML en el servidor (Ctrl+F8) de alto rendimiento** también aparece disponible en el ayudante de entrada del proyecto. Haga clic con el botón derecho del ratón sobre el proyecto, una carpeta o un archivo y seleccione **Validar documento XML en el servidor** para validar los datos XML o XBRL en el objeto seleccionado.

Observación: La validación de Raptor se encuentra disponible en la vista Texto, Cuadrícula y XBRL.

10.5 Menú XSL/XQuery

El lenguaje de transformación XSL permite especificar cómo se debe convertir un documento XML en otros documentos XML o archivos de texto. Un tipo de documento XML que se genera con un documento XSLT es el documento FO, que más adelante se puede procesar para generar archivos PDF. Authentic Desktop contiene procesadores XSLT integrados (para XSLT 1.0, 2.0 y 3.0) y puede conectarse a un procesador FO del sistema del usuario para transformar archivos XML y generar archivos en diferentes formatos de salida. La ubicación del procesador FO debe indicarse en la pestaña *XSL* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**) para que Authentic Desktop pueda utilizarlo.

Los comandos necesarios para realizar estas transformaciones están disponibles en el menú **XSL/XQuery**. Además, este menú incluye los comandos necesarios para trabajar con el depurador XSLT/XQuery de Altova.



© 2015 Altova GmbH

10.5.1 Transformación XSL

F10

El comando **XSL/XQuery | Transformación XSL** transforma un documento XML usando una hoja de estilos XSLT. La transformación se puede realizar usando el motor XSLT integrado de Altova (el motor XSLT 1.0 para las hojas de estilos XSLT 1.0, el motor XSLT 2.0 para las hojas de estilos XSLT 2.0 y el motor XSLT 3.0 para hojas de estilos XSLT 3.0), el módulo MSXML de Microsoft o un procesador XSLT externo. El procesador que se usa junto con este comando se especifica en la pestaña *XSL* del cuadro de diálogo Opciones (**Herramientas | Opciones**).

Si el documento XML contiene una referencia a una hoja de estilos XSLT, esta hoja de estilos se usa para la transformación. (Si el documento forma parte de un proyecto, puede asignar una hoja de estilos XSLT a cada carpeta del proyecto, usando el cuadro de diálogo Propiedades. Haga clic con el botón secundario en la carpeta de proyecto o en el archivo que desea transformar y seleccione el comando **Transformación XSL**.) Si no se asignó ninguna hoja de estilos al archivo XML, aparece un mensaje solicitando la hoja de estilos XSLT que se debe usar. También puede seleccionar un archivo mediante un recurso global o una URL (haga clic en el botón **Examinar**) o un archivo de las ventanas Authentic Desktop que están abiertas (haga clic en el botón **Ventana**).

Automatizar transformaciones XSLT con RaptorXML

RaptorXML es una aplicación independiente de Altova para validar XML, transformar XSLT y ejecutar XQuery. Se puede usar desde la línea de comandos, por una interfaz COM, en programas Java y en aplicaciones .NET. Por tanto, puede automatizar las tareas de transformación XSLT con RaptorXML. Por ejemplo, puede crear un archivo de procesamiento por lotes que llame a RaptorXML para transformar un grupo de documentos. Para más información, consulte la documentación de RaptorXML.

Transformaciones con archivos ZIP como destino

Para desviar los documentos de salida de las transformaciones a un archivo ZIP (incluidos los documentos OOXML, como `.docx`, por ejemplo) es necesario especificar el protocolo ZIP en la ruta de acceso del archivo de salida. Por ejemplo:

```
nombreArchivo.zip|zip/nombreArchivo.xxx
nombreArchivo.docx|zip/nombreArchivo.xxx
```

Nota: puede que sea necesario crear la estructura de directorios antes de ejecutar la transformación. Si está generando archivos para un archivo Open Office XML, es necesario comprimir los ficheros del archivo para poder crear el archivo OOXML de nivel superior (por ejemplo, `.docx`).

10.5.2 Transformación XSL-FO

Ctrl+F10

FO es un formato XML que describe documentos paginados. Los procesadores FO, como el procesador FOP de Apache XML Project, toman un archivo FO de entrada y generan un archivo PDF de salida. Esto significa que el proceso para generar documentos PDF a partir de documentos XML tiene dos fases:

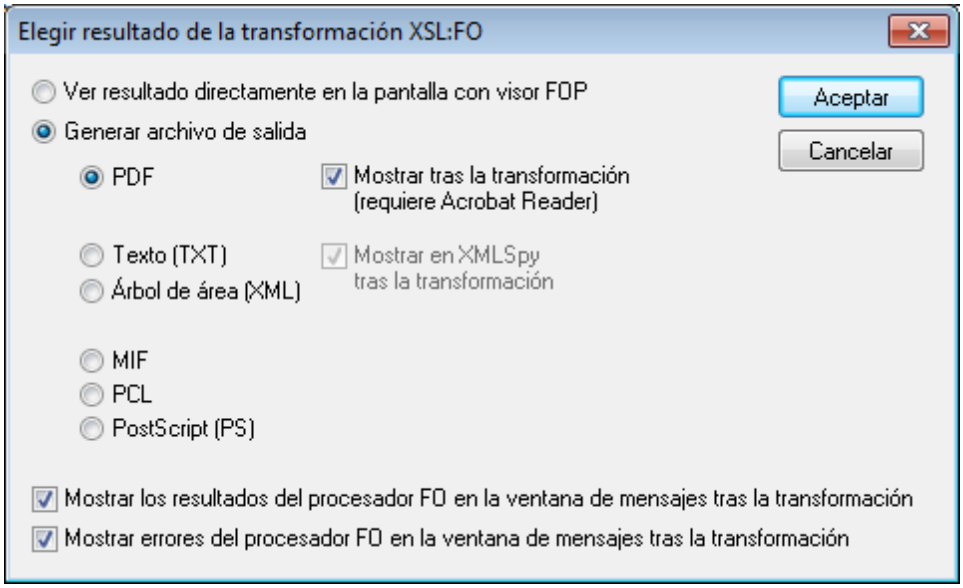
1. Primero el documento XML se transforma en un documento FO usando una hoja de estilos XSLT.
2. Después el procesador FO procesa el documento FO y genera un archivo PDF.

El comando **XSL/XQuery | Transformación XSL-FO** transforma un documento XML o un documento FO en PDF.

- Si ejecuta el comando **Transformación XSL-FO** en un documento XML de origen, entonces se ejecutan los dos pasos descritos anteriormente en secuencia, uno después del otro. Si en el documento XML no hay una referencia a la hoja de estilos XSLT necesaria para transformarlo en documento FO, la aplicación solicita que le asigne una hoja de estilos. Recuerde que también puede seleccionar un archivo mediante un recurso global o una URL (botón **Examinar**) o seleccionar uno de los archivos que están abiertos en las ventanas de Authentic Desktop (botón **Ventana**). La transformación del documento XML en XSL-FO la realiza el procesador XSLT que se indica en la pestaña *XSL* en el cuadro de diálogo Opciones (**Herramientas | Opciones**). El procesador XSLT predeterminado es el procesador XSLT integrado de Authentic Desktop. El documento FO resultante lo procesa el procesador FO que se indica en la pestaña *XSL* en el cuadro de diálogo Opciones (**Herramientas | Opciones**).
- Si el comando **Transformación XSL-FO** se ejecuta en un documento FO, el documento se procesa con el procesador FO indicado en la pestaña *XSL* en el cuadro de diálogo Opciones (**Herramientas | Opciones**).

Resultado de la transformación XSL:FO
Al hacer clic en el comando **Transformación XSL-FO** aparece el cuadro de diálogo "Elegir resultado de la transformación XSL:FO" (*imagen siguiente*). Recuerde que si el documento activo es un documento XML que no tiene asignado ninguna hoja de estilos XSLT, XMLSpy primero le pide un archivo XSLT.

© 2015 Altova GmbH


Elegir resultado de la transformación XSL:FO
Ver resultado directamente en la pantalla con visor FOP
Generar archivo de salida
PDF
Mostrar tras la transformación (requiere Acrobat Reader)
Texto (TXT)
Árbol de área (XML)
Mostrar en XMLSpy tras la transformación
MIF
PCL
PostScript (PS)
Aceptar
Cancelar
Mostrar los resultados del procesador FO en la ventana de mensajes tras la transformación
Mostrar errores del procesador FO en la ventana de mensajes tras la transformación

El resultado generado por el procesador FO se puede ver directamente en pantalla usando un visor FOP o puede generar un archivo de salida en uno de estos formatos: PDF, texto, árbol de área, MIF, PCL o PostScript. Si lo desea, los mensajes del procesador FO y los mensajes de error del procesador pueden aparecer en la ventana Mensajes. Para activar estas opciones marque las casillas correspondientes situadas al final del cuadro de diálogo.

Nota:

- A no ser que desactive la casilla correspondiente en el programa de instalación de Authentic Desktop, el procesador FOP de Apache XML Project se instalará en la carpeta `C:\ProgramData\Altova\SharedBetweenVersions`. Y si el procesador está instalado, su ruta de acceso aparecerá automáticamente en la pestaña *XSL* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**). Si quiere puede cambiar esta ruta de acceso para usar otro procesador FO distinto.
- El comando **Transformación XSL-FO** no solamente se puede usar con el archivo activo en la ventana principal. También puede usarlo con cualquier archivo o carpeta del proyecto activo. Para ello, seleccione los archivos o carpetas en la ventana Proyecto, haga clic con el botón secundario y seleccione el comando **Transformación XSL-FO**. Para la transformación se usa la hoja de estilos XSLT asignada a la carpeta de proyecto seleccionada.

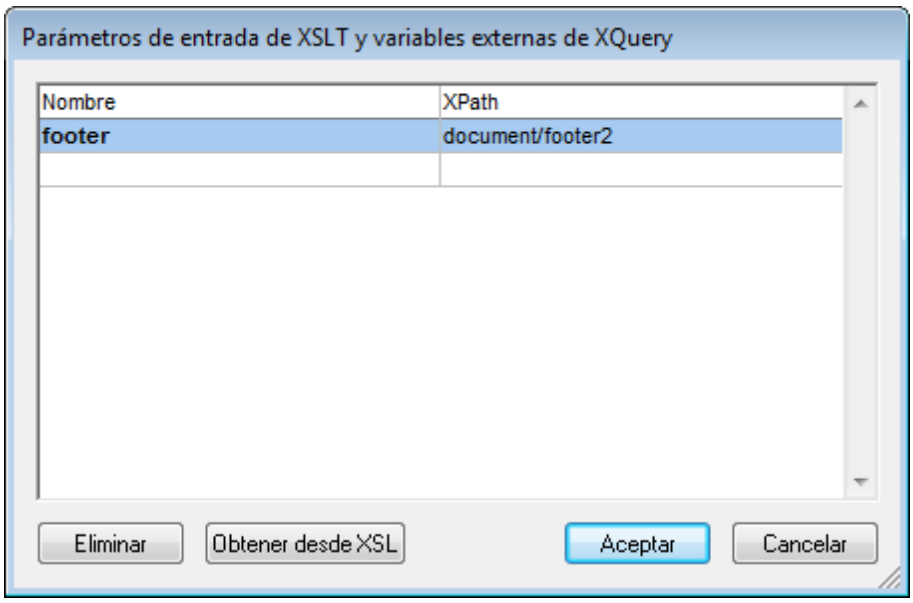
10.5.3 Parámetros de XSL / Variables de XQuery

El comando **XSL/XQuery | Parámetros de XSL / Variables de XQuery** abre el cuadro de diálogo "Parámetros de entrada de XSL y variables externas de XQuery" (*imagen siguiente*). Aquí puede escribir el nombre de los parámetros que desea pasar a la hoja de estilos XSLT y las variables externas de XQuery que desea pasar al documento XQuery, así como sus correspondientes valores. En Authentic Desktop estos parámetros se usan de esta manera:

- Cuando se usa el comando **XSL/XQuery | Transformación XSL** para transformar un documento XML, los valores del parámetros guardados en este cuadro de diálogo se pasan al archivo XSLT seleccionado y utilizado para la transformación.
- Cuando se usa el comando **XSL/XQuery | Ejecutar XQuery** para procesar un documento XQuery, los valores de las variables externas de XQuery guardadas en este cuadro de diálogo se pasan al documento XQuery para la ejecución.

Nota: los parámetros y las variables que introduzca en este cuadro de diálogo se pasan solamente al motor XSLT integrado de Altova. Por tanto, si usa MSXML o un motor externo, estos parámetros no se pasan al motor de procesamiento.

Cómo usar los parámetros de XSLT
El valor del parámetro puede ser una expresión XPath sin comillas o una cadena de texto delimitada por comillas. Si el documento activo es un archivo XSLT, entonces está habilitado el botón **Obtener desde XSL**. Al hacer clic en este botón se insertan en el cuadro de diálogo los parámetros declarados en el archivo XSLT y sus valores predeterminados. Es decir, este botón le permite incluir rápidamente los parámetros declarados y después cambiar sus valores predeterminados si lo necesita.



Nota importante: una vez introducidos en este cuadro de diálogo, los pares de parámetros y valores se usan para todas las transformaciones subsiguientes hasta que se eliminen del cuadro de diálogo o se reinicie la aplicación. Los parámetros introducidos en este cuadro de diálogo se definen para toda la aplicación y se pasan al documento XSLT correspondiente en cada transformación que se realice a partir de ese momento a través del entorno IDE. En otras palabras:

© 2015 Altova GmbH

- los parámetros no están asociados a ningún tipo de documento en particular
- los parámetros introducidos en el cuadro de diálogo "Parámetros de entrada de XSL y variables externas de XQuery" se borran cuando se cierra Authentic Desktop.

Ejemplo de uso de parámetros de XSLT
En este ejemplo, seleccionamos en el documento XML un pie de página (`footer`) para el documento, entre tres pies de página posibles (`footer1`, `footer2`, `footer3`).

```
<?xml version="1.0" encoding="UTF-8"?>
<document xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="C:\workarea\footers\footers.xsd">
   <footer1>Footer 1</footer1>
   <footer2>Footer 2</footer2>
   <footer3>Footer 3</footer3>
   <title>Document Title</title>
   <para>Paragraph text.</para>
   <para>Paragraph text.</para>
</document>
```

El archivo XSLT contiene un parámetro local llamado `footer` en la plantilla del elemento raíz. El valor predeterminado de este parámetro es `footer1`. Después se crea una instancia del valor del parámetro en la plantilla con el valor `$footer` en la definición del bloque `footer`.

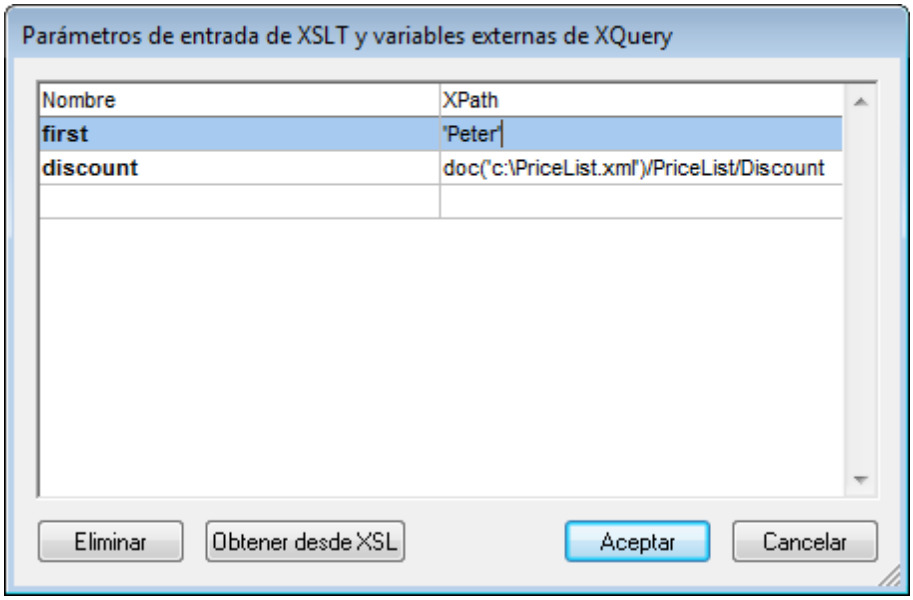
```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:fo="http://www.w3.org/1999/XSL/Format">
   ...
   <xsl:param name="footer" select="document/footer1" />
   ...
   <xsl:template match="/">
      <fo:root>
         <xsl:copy-of select="$fo:layout-master-set" />
         <fo:page-sequence master-reference="default-page"
            initial-page-number="1" format="1">
            <fo:static-content flow-name="xsl-region-after"
               display-align="after">
               ...
               <fo:inline color="#800000" font-size="10pt" font-weight="bold">
                  <xsl:value-of select="$footer"/>
               </fo:inline>
               ...
            </fo:static-content>
         </fo:page-sequence>
      </fo:root>
   </xsl:template>
</xsl:stylesheet>
```

En el cuadro de diálogo "Parámetros de entrada de XSLT" se puede introducir un valor nuevo para el parámetro `footer`, como por ejemplo la XPath `document/footer2` (*imagen anterior*) o una cadena de texto. Durante la transformación, este valor se pasa al parámetro `footer` de la plantilla para el elemento raíz y es el valor utilizado cuando se crea la instancia del bloque `footer`.

Nota:

- Si usa el comando **XSL/XQuery | Transformación XSL-FO**, los parámetros introducidos en el cuadro de diálogo "Parámetros de entrada de XSLT y variables externas de XQuery" **no se pasan** a la hoja de estilos. Para que estos parámetros se usen en el archivo PDF de salida, antes es necesario transformar el documento XML en FO con el comando **XSL/XQuery | Transformación XSL**) y después transformar el FO en PDF con el comando **XSL/XQuery | Transformación XSL-FO**.
- Si usa un procesador XSLT que no sea uno de los motores XSLT integrados de Altova, los parámetros que introduzca en el diálogo "Parámetros de entrada de XSLT y variables externas de XQuery" no se pasan al procesador externo.

Cómo usar las variables externas de XQuery
El valor de las variables externas de XQuery puede ser una expresión XPath sin comillas o una cadena de texto delimitada por comillas. El tipo de datos de la variable externa se especifica en la declaración de variable en el documento XQuery.



Nota: una vez introducidas en este cuadro de diálogo, las variables externas de XQuery se utilizan para todas las ejecuciones subsiguientes hasta que se borren del cuadro de diálogo o se reinicie la aplicación. Las variables introducidas en este cuadro de diálogo se definen para toda la aplicación y se pasan al documento XQuery correspondiente en cada ejecución que se realice a partir de ese momento a través del entorno IDE. En otras palabras:

- las variables no están asociadas a ningún tipo de documento en particular
- las variables introducidas en el cuadro de diálogo "Parámetros de entrada de XSL y variables externas de XQuery" se borran cuando se cierra Authentic Desktop.

Ejemplo de uso de variables externas de XQuery
En este ejemplo, en el documento XQuery se declaró una variable `$first` que se utiliza en la cláusula `return` de la expresión FLWOR:

```
xquery version "1.0";
```

 © 2015 Altova GmbH

```
declare variable $first as xs:string external;
let $last := "Jones"
return concat($first, " ", $last )
```

Este XQuery devuelve `Peter Jones`, si el valor de la variable externa (introducida en el cuadro de diálogo "Parámetros de entrada de XSL y variables externas de XQuery") es `Peter`. Es necesario tener en cuenta que:

- La palabra clave `external` de la declaración de variable del documento XQuery indica que esta variable es una variable externa.
- Definir el tipo estático de la variable es opcional. Si no se indica un tipo de datos para la variable en la declaración de variable, entonces al valor de la variable se le asigna el tipo `xs:untypedAtomic`.
- Si se declara una variable externa en el documento XQuery, pero no se pasa al documento XQuery ninguna variable externa que tenga ese mismo nombre, entonces se emite un mensaje de error.
- Si se declara y se introduce una variable externa en el cuadro de diálogo "Parámetros de entrada de XSL y variables externas de XQuery", entonces se considera que está en el ámbito del documento XQuery que se está ejecutando. Si se declara una variable nueva con el mismo nombre dentro del documento XQuery, la variable nueva reemplaza temporalmente a la variable externa. Por ejemplo, el XQuery que aparece a continuación devuelve `Paul Jones` aunque la variable externa `$first` que está en el ámbito del documento tiene el valor `Peter`.

  ```
  xquery version "1.0";
  declare variable $first as xs:string external;
  let $first := "Paul"
  let $last := "Jones"
  return concat($first, " ", $last )
  ```

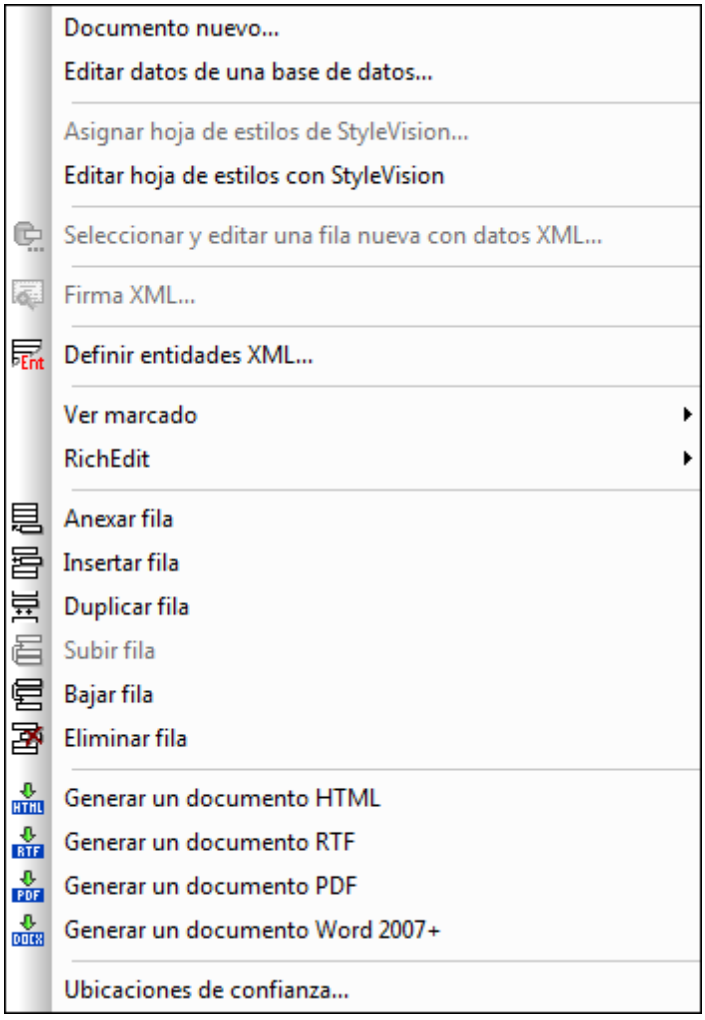
Nota: no es erróneo definir una variable externa de XQuery (o un parámetro de XSLT) en el diálogo "Parámetros de entrada de XSL y variables externas de XQuery" y después no utilizarla en el documento XQuery. Tampoco es erróneo definir un parámetro de XSLT (o una variable externa de XQuery) en este cuadro de diálogo y después no usarlo en una transformación XSLT.

10.6 Menú Authentic

En la vista Authentic puede editar documentos XML basados en **archivos SPS creados con Altova StyleVision.** Los archivos SPS son hojas de estilos que permiten mostrar el archivo XML de forma gráfica en la vista Authentic. Además de contener información sobre la presentación, el archivo SPS también permite escribir datos en el archivo XML. Estos datos se procesan de forma dinámica usando las funciones de las hojas de estilos XSLT y su resultado se reproduce instantáneamente en la vista Authentic.

También puede usar archivos SPS para mostrar una vista XML editable de una base de datos. El archivo SPS contiene información para conectarse a la base de datos, mostrar los datos de la base de datos en la vista Authentic y volver a escribir datos en la base de datos.

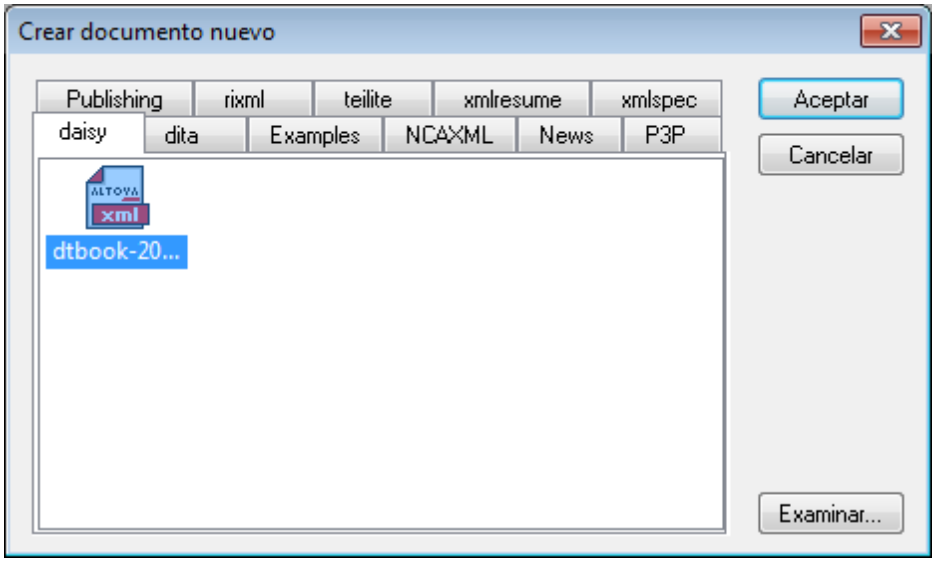
El menú **Authentic** incluye todos los comandos necesarios para editar documentos XML en la vista Authentic. Para aprender a usar la vista Authentic, consulte los tutoriales de la vista Authentic.



 © 2015 Altova GmbH

10.6.1 Documento nuevo

Este comando sirve para abrir una plantilla de documento XML nuevo en la vista Authentic. La plantilla de documento XML está basada en un archivo SPS (StyleVision Power Stylesheet) y se abre seleccionando el archivo SPS en el cuadro de diálogo "Crear documento nuevo" (*imagen siguiente*). Tras seleccionar el archivo SPS haga clic en **Aceptar**. La plantilla de documento XML definida para el archivo SPS elegido se abre en la vista Authentic.



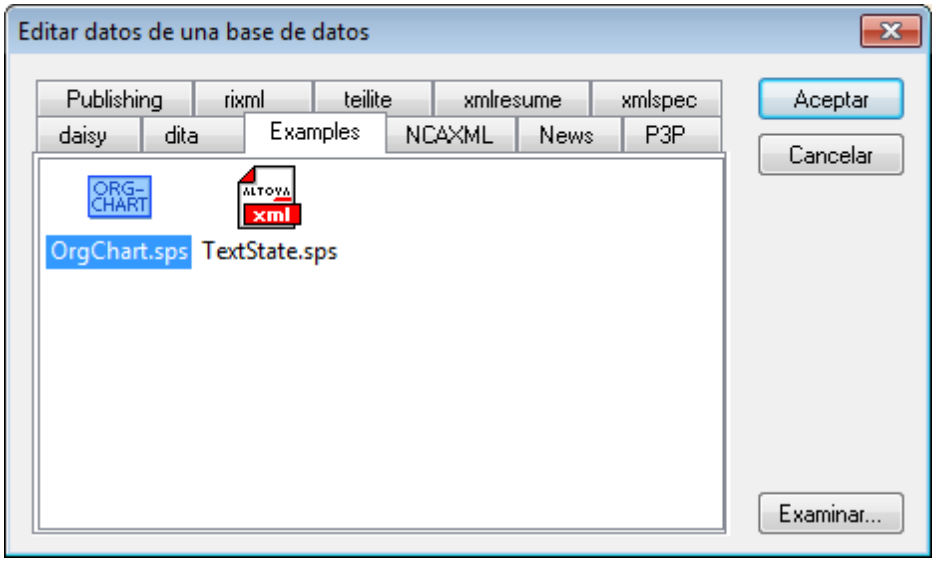
El cuadro de diálogo "Crear documento nuevo" ofrece varias plantillas de documento XML basadas en esquemas y archivos DTD frecuentes. Si lo prefiere, también puede buscar archivos SPS personalizados que tengan asignado un archivo XML. Los archivos SPS se crean con Altova StyleVision, una herramienta con la que puede diseñar plantillas de documento XML a partir de un esquema XML o un archivo DTD. Tras diseñar el archivo SPS en StyleVision puede asignarle un archivo XML como plantilla XML. Los datos de esta plantilla XML aportan los datos de partida de la plantilla de documento que se abre en la vista Authentic de Authentic Desktop.

La nueva plantilla de documento XML las propiedades de presentación definidas en el archivo SPS y los datos del archivo XML que seleccionó como plantilla XML. El usuario de la vista Authentic puede editar la plantilla de documento XML en una interfaz gráfica WYSIWYG y guardar los cambios en un archivo XML.

10.6.2 Editar datos de una base de datos

El comando **Authentic** | **Editar datos de una base de datos** sirve para abrir una vista editable de una base de datos (BD) en la vista Authentic. El archivo SPS contiene la información necesaria para conectarse a la BD y sobre cómo presentar la BD y aceptar los cambios realizados en ella en la vista Authentic. El archivo SPS basado en la BD se abre con el comando **Editar datos de una base de datos...**. Al abrirlo se establece una conexión a la BD y sus datos se representan (a través de la lente de XML) en la vista Authentic.

Al hacer clic en el comando **Editar datos de una base de datos...** aparece un cuadro de diálogo del mismo nombre.



Busque el archivo SPS que desea usar, selecciónelo y haga clic en **Aceptar**. A continuación se establece la conexión con la BD y en la vista Authentic se abre una vista editable de la BD. El diseño de la vista de la BD que aparece en la vista Authentic se define dentro del archivo SPS.

Nota: si con el comando **Editar datos de una base de datos** intenta abrir un archivo SPS que no está basado en una BD o si intenta abrir un archivo SPS basado en una BD que se creó con una versión de StyleVision anterior a StyleVision 2005, entonces recibirá un error.

Nota: los archivos SPS se crean con Altova StyleVision.

© 2015 Altova GmbH

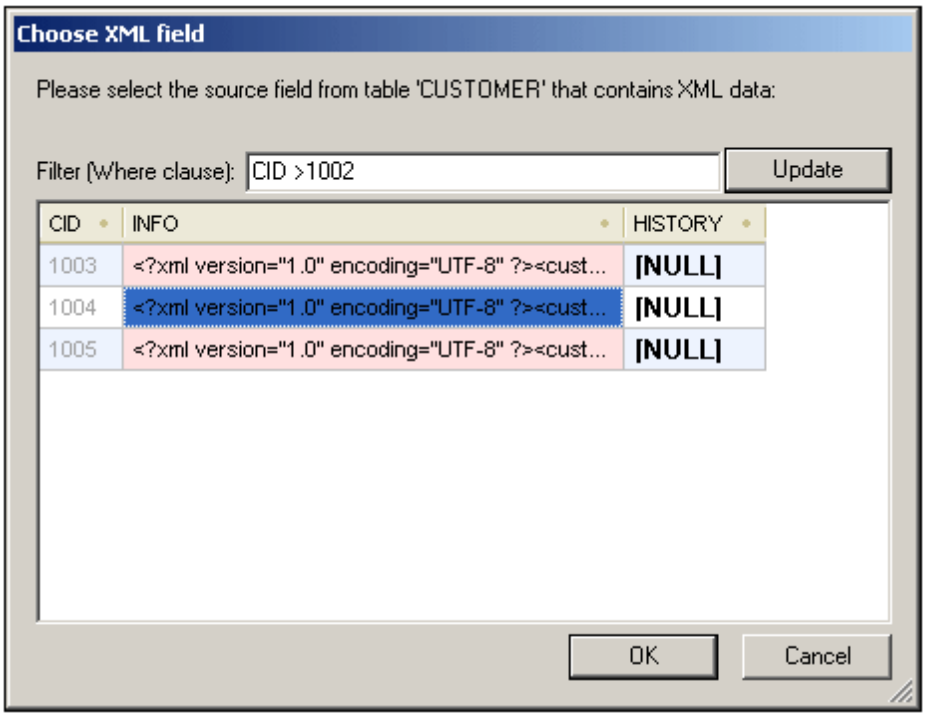
10.6.3 Editar una hoja de estilos de StyleVision

El comando **Authentic | Editar una hoja de estilos de StyleVision** inicia la aplicación StyleVision para que pueda editar el archivo SPS seleccionado. Este comando solamente se habilita si el documento XML activo tiene asignado un archivo SPS.

10.6.4 Seleccionar y editar una fila nueva con datos XML

El comando **Seleccionar y editar una fila nueva con datos XML** sirve para seleccionar una fila nueva de la tabla pertinente en una BD XML, como IBM DB2. Esta fila aparece en la vista Authentic, donde se puede editar y guardar con los cambios realizados.

Cuando se usa una BD XML como origen de datos XML, los datos XML que se presentan en la vista Authentic corresponden al documento XML que está en una de las celdas de la columna de datos XML. El comando **Seleccionar y editar una fila nueva con datos XML** permite seleccionar un documento XML de otra celda (o fila) de esa columna XML. Al seleccionar este comando aparece el cuadro de diálogo "Elegir campo XML" (*imagen siguiente*), que muestra la tabla en la que está la columna XML.



En este cuadro de diálogo puede introducir un filtro para filtrar la tabla. Este filtro debe ser una cláusula `WHERE` SQL (solamente la condición, sin la palabra clave `WHERE`. Por ejemplo `CID>1002`). Haga clic en **Actualizar** para actualizar el cuadro de diálogo. En la imagen anterior puede ver la tabla después de aplicar el filtro. El siguiente paso es seleccionar la celda que contiene el documento XML. Después haga clic en **Aceptar** y el documento XML de la celda (fila) seleccionada se carga en la vista Authentic.

© 2015 Altova GmbH

10.6.5 Definir entidades XML

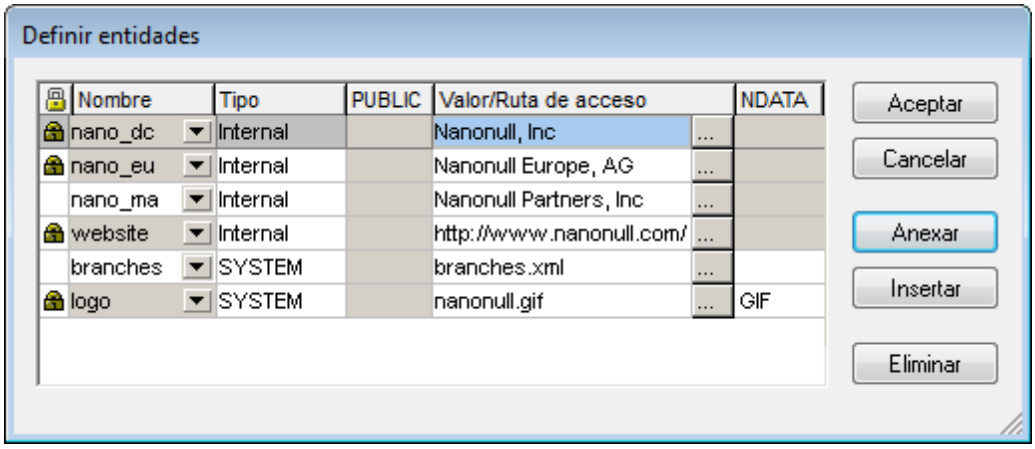
Tanto si su documento está basado en una DTD como si se basa en un esquema XML, puede definir las entidades que se deben usar en la vista Authentic. Una vez definidas, estas entidades aparecen en el ayudante de entrada Entidades y en el submenú **Insertar entidad** del menú contextual. Al hacer doble clic en una entidad del ayudante de entrada Entidades, esta entidad se inserta en el punto de inserción del cursor.

Las entidades son útiles cuando hay una cadena de texto, un fragmento XML o un recurso externo que se va a utilizar en varios puntos del documento. La entidad (básicamente un nombre corto que representa los datos) se define en el cuadro de diálogo "Definir entidades". Después de definirla, puede usar la entidad en varios puntos del documento. Así puede ahorrar tiempo y mantener el documento más fácilmente.

En un documento puede usar dos tipos de entidades: **entidades analizadas**, es decir, datos XML (una cadena de texto o un fragmento de un documento XML) o **entidades sin analizar**, es decir, datos no XML como un archivo binario (por lo general, objetos gráficos, sonoros o multimedia). Cada entidad tiene un nombre y un valor. En el caso de las entidades analizadas, la entidad es un marcador de posición para los datos XML. El valor de la entidad es los datos XML propiamente dichos o un URI que apunta a un archivo XML que contiene los datos XML. En el caso de las entidades sin analizar, el valor de la entidad es un URI que apunta al archivo de datos no XML.

Siga estos pasos para definir una entidad:

1. Haga clic en **Authentic | Definir entidades XML...** y aparece el cuadro de diálogo "Definir entidades".



2. Escriba el nombre de su entidad en el campo **Nombre**. Se trata del nombre que aparecerá en el ayudante de entrada Entidades.
3. En la lista desplegable del campo **Tipo** seleccione el tipo de entidad. Hay tres tipos. Una entidad interna (`Internal`) es una cuyo texto se almacena en el documento XML propiamente dicho. Las opciones **`PUBLIC`** y **`SYSTEM`** indican que el recurso está fuera del XML y que se localizará usando un identificador público o de sistema. Un identificador de sistema es un URI que ofrece la ubicación del recurso. Un identificador público es un identificador que no depende de la ruta de acceso y permite a algunos procesadores identificar el recurso. Si se indica tanto un identificador público como de sistema, el identificador público se resuelve como identificador de sistema y se usa el identificador de sistema.
4. Si selecciona el tipo `PUBLIC`, escriba el identificador público de su recurso en el campo

PUBLIC. Si selecciona el tipo Internal o SYSTEM, se deshabilita el campo PUBLIC.

5. En el campo **Valor/Ruta de acceso** tiene estas opciones:
   - Si el tipo de entidad es **Internal**, escriba la cadena de texto de la entidad. La cadena de texto no puede estar delimitada por comillas. Si escribe comillas, se entenderá que forman parte de la cadena de texto.
   - Si el tipo de entidad es **SYSTEM**, escriba el URI del recurso o seleccione un recurso de su red local con el botón **Examinar**. Si el recurso contiene datos analizados, este debe ser un archivo XML (es decir, debe tener la extensión .xml). El recurso también puede ser un archivo binario, como un archivo GIF.
   - Si el tipo de entidad es **PUBLIC**, debe introducir también un identificador de sistema en este campo.
6. El campo **NDATA** le dice al procesador que la entidad no se debe analizar sino que se debe enviar al procesador adecuado. Por tanto, rellene el campo NDATA solamente si la entidad es una entidad sin analizar.

Características del cuadro de diálogo

En este cuadro de diálogo puede anexar, insertar y eliminar entidades haciendo clic en los botones correspondientes. También puede ordenar las entidades por orden alfabético en base al valor de cualquiera de las columnas. Al hacer clic una vez en el encabezado de la columna, la columna se ordena alfabéticamente en orden ascendente. Si hace clic dos veces en el encabezado, la columna se ordena por orden descendiente. También puede ajustar el tamaño del cuadro de diálogo y el ancho de las columnas.

Una vez utilizadas en el documento XML, las entidades se bloquean y no se pueden editar en el cuadro de diálogo "Definir entidades". Las entidades bloqueadas se señalizan con un símbolo en forma de candado junto a la primera columna del diálogo. Así se garantiza la validez del documento (el documento no es válido si se hace referencia a una entidad que no se definió).

También recibirá una advertencia si define la misma entidad dos veces.

Limitaciones de las entidades

- Una entidad que está dentro de otra entidad no se resuelve, ni en el cuadro de diálogo, ni en la vista Authentic ni en el resultado XSLT. Además, el carácter & de dichas entidades se presenta con caracteres de escape, es decir, &.
- Las entidades externas no se resuelven en la vista Authentic, excepto cuando la entidad es un archivo de imagen y se introduce como valor de un atributo que está definido en el esquema como atributo de tipo ENTITY o ENTITIES. Este tipo de entidades se resuelven cuando el documento se procesa con un XSLT generado a partir del archivo SPS.

© 2015 Altova GmbH

10.6.6 Ver marcado

El comando **Ver marcado** tiene un submenú con opciones para controlar el marcado del documento XML en la vista Authentic. A continuación describimos estas opciones.

El comando **Ocultar marcado** oculta los símbolos de marcado del documento en la vista Authentic.

El comando **Mostrar marcado en tamaño pequeño** muestra símbolos pequeños de marcado en la vista Authentic.

El comando **Mostrar marcado en tamaño grande** muestra símbolos grandes de marcado en la vista Authentic.

El comando **Mostrar marcado mixto** muestra símbolos de marcado de tamaño mixto en la vista Authentic. El diseñador del archivo SPS puede definir diferentes tamaños de marcado para cada elemento y atributo del documento. En el modo de marcado mixto, el usuario de la vista Authentic utiliza este tipo personalizado de marcado.

10.6.7 Anexar/insertar/duplicar/eliminar fila

El comando **Anexar fila** anexa una fila a la tabla seleccionada en la vista Authentic.

El comando **Insertar fila** inserta una fila en la tabla seleccionada en la vista Authentic.

El comando **Duplicar fila** crea una copia de la fila seleccionada en la tabla de la vista Authentic.

El comando **Eliminar fila** elimina la fila seleccionada en la tabla de la vista Authentic.

© 2015 Altova GmbH

10.6.8 Subir/bajar fila

En una tabla de la vista Authentic el comando **Subir fila** sube la fila seleccionada una fila más arriba.

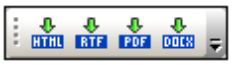
En una tabla de la vista Authentic el comando **Bajar fila** baja la fila seleccionada una fila más abajo.

10.6.9 Generar documento HTML, RTF, PDF o Word 2007+

Estos cuatro comandos sirven para generar documentos de salida a partir del documento XML de un archivo PXF en la vista Authentic:

- **Generar un documento HTML**
- **Generar un documento RTF**
- **Generar un documento PDF**
- **Generar un documento Word 2007+**

Estos cuatro comandos también están disponibles en la barra de herramientas "Formulario XML Portátil" (*imagen siguiente*).



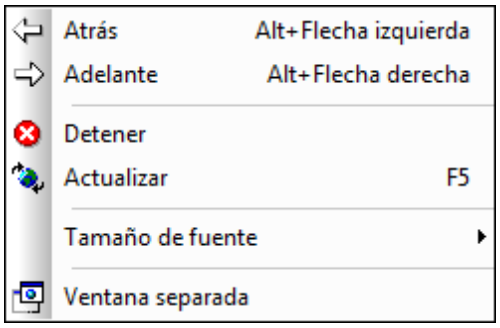
Al hacer clic en estos comandos o botones se genera un documento HTML, RTF, PDF o DocX, dependiendo del comando seleccionado.

Estos comandos se habilitan solamente si el archivo abierto en la vista Authentic es un archivo PXF. Además, si el archivo PXF contiene una hoja de estilos para un formato de salida concreto, solamente se habilita el comando correspondiente. Por ejemplo, si el archivo PXF contiene hojas de estilos XSLT para HTML y RTF, entonces solamente se habilitan los comandos para generar documentos HTML y RTF, mientras que los comandos para PDF y DocX se deshabilitan.

© 2015 Altova GmbH

10.7 Menú Explorador

Los comandos del menú **Explorador** están disponibles solamente en la vista Explorador. No obstante, los comandos **Adelante** y **Atrás** están disponibles también en la vista Esquema, donde sirven para pasar a comandos usados previamente.



10.7.1 Atrás

El comando Atrás (**Alt + flecha izda**) está disponible en las vistas Explorador y Esquema.

En la vista Explorador el comando **Atrás** muestra la página que visualizó en el paso anterior. También puede usar la tecla **Retroceso**. Este comando es muy práctico si hace clic en un enlace del documento XML y quiere volver al documento.

En la vista Esquema, el comando **Atrás** le lleva al componente o vista que visualizó en el paso anterior. Con este comando puede retroceder y volver a ver un máximo de 500 posiciones.

© 2015 Altova GmbH

10.7.2 Adelante

El comando **Adelante** (**Alt + flecha der)** está disponible en la vista Explorador. También está disponible en la vista Esquema si ya utilizó el comando **Atrás**. El comando **Adelante** le permite avanzar por (i) páginas vistas previamente en la vista Explorador y (ii) vistas previas de componentes en la vista Esquema.

10.7.3 Detener

El comando **Detener** está disponible en la vista Explorador y le ordena que deje de cargar el documento. Esto puede ser práctico si está descargando archivos externos o gráficos de gran tamaño con una conexión a Internet lenta y quiere interrumpir el proceso.

© 2015 Altova GmbH

10.7.4 Actualizar

El comando **Actualizar (F5)** está disponible en la vista Explorador y sirve para actualizar la vista, es decir, se vuelve a cargar el documento y su documentos relacionados (las hojas de estilos CSS/XSL, los documentos DTD, etc.).

10.7.5 Fuentes

Al hacer clic en el comando **Fuentes** aparece un submenú donde puede seleccionar el tamaño de fuente predeterminado en el que se representa el texto del documento XML. Este comando está disponible en la vista Explorador.

© 2015 Altova GmbH

10.7.6 Ventana separada

El comando **Ventana separada** está disponible en la vista Explorador y abre la vista Explorador en otra ventana diferente, separada de las demás vistas. Así puede ver la vista Explorador en paralelo a una vista de edición, por ejemplo.

Si tras realizar cambios en la vista de edición desea actualizar la ventana separada de la vista Explorador, pulse **F5** en la vista de edición. Si desea volver a acoplar la ventana separada en la interfaz, haga clic en la ventana de la vista Explorador y después en el comando **Ventana separada**.

10.8 Menú Herramientas

Con el menú **Herramientas** puede:

- Revisar la ortografía de sus documentos XML
- Comparar dos archivos y ver las diferencias que hay entre ellos.
- Comparar dos carpetas y ver las diferencias que hay entre ellas.
- Definir recursos globales
- Cambiar la configuración activa para los recursos globales de Authentic Desktop
- Personalizar su versión de Authentic Desktop: definiendo sus propias barras de herramientas, teclas de acceso rápido, macros y menús.
- Definir opciones de configuración globales de Authentic Desktop

© 2015 Altova GmbH

10.8.1 Ortografía

El corrector ortográfico de Authentic Desktop (y sus diccionarios integrados)en la vista Authentic.

Nota: los diccionarios integrados que vienen con el software de Altova no son un reflejo de las preferencias lingüísticas de Altova. Los diccionarios integrados del corrector ortográfico se seleccionaron porque su licencia permite su distribución con software comercial (MPL, LGPL o BSD, por ejemplo). Existen muchos otros diccionarios de código abierto, pero se distribuyen con licencias más restrictivas, como la licencia GPL. Estos diccionarios pueden descargarse e instalarse por separado desde la página http://www.altova.com/es/dictionaries. Recuerde que es decisión suya aceptar o no las condiciones de uso de la licencia del diccionario elegido. También es responsabilidad suya comprobar si el diccionario puede utilizarse en su equipo o no.

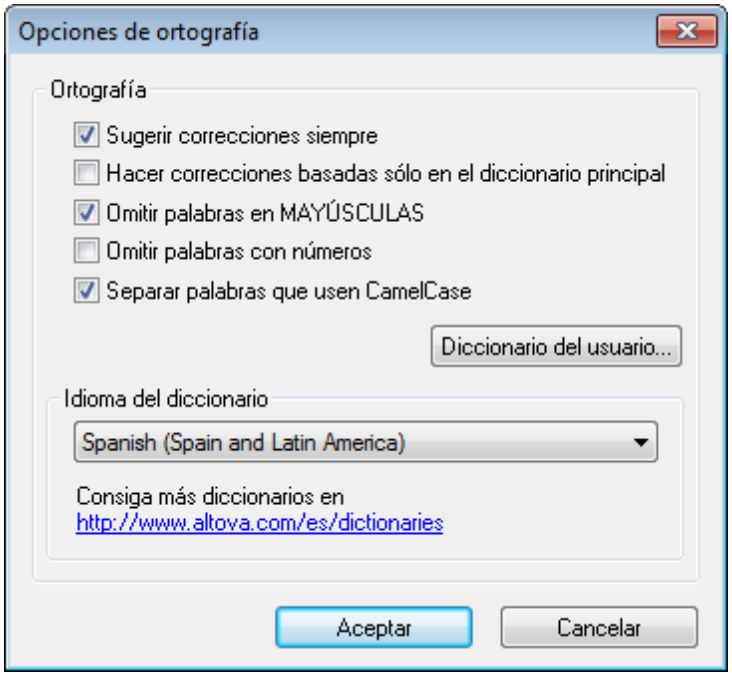
Este apartado describe el uso del corrector ortográfico y tiene tres partes:

- Seleccionar el idioma del corrector ortográfico
- Ejecutar la revisión ortográfica

Seleccionar el idioma del corrector ortográfico

Siga estos pasos para seleccionar el idioma del corrector ortográfico:

1. Haga clic en el comando de menú **Herramientas | Opciones de ortografía**.
2. Aparece el cuadro de diálogo "Opciones de ortografía XML". Haga clic en el botón **Más opciones de ortografía**.
3. Aparece el cuadro de diálogo "Opciones de ortografía" (*imagen siguiente*). Seleccione un idioma en el cuadro combinado *Idioma del diccionario*.



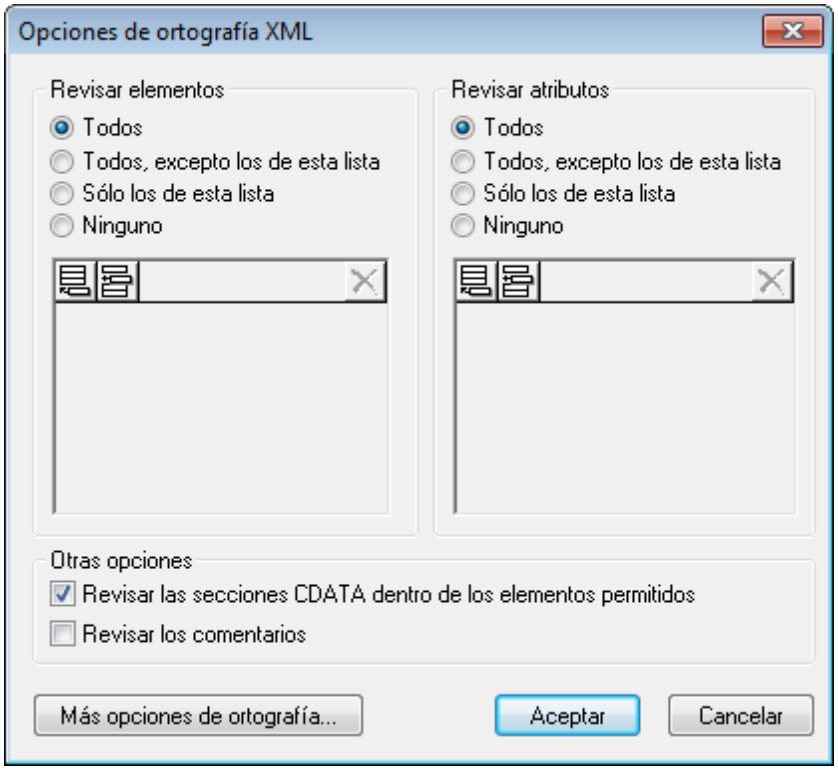
4. Para terminar haga clic en **Aceptar**.

El idioma del diccionario seleccionado será utilizado durante la revisión ortográfica. Si el idioma que desea utilizar no está instalado, puede descargar más diccionarios desde el sitio web de

Altova. Para más información consulte el apartado Opciones de ortografía (añadir diccionarios para el corrector ortográfico).

Definir el contexto de la revisión ortográfica

Cuando el corrector ortográfico se utiliza en la vista Texto o Cuadrícula, puede definir el contexto de la revisión antes de ejecutar el corrector. Para ello seleccione el comando **Herramientas | Opciones de ortografía** y defina el contexto que desea utilizar en el cuadro de diálogo Opciones de ortografía XML que aparece (*imagen siguiente*).

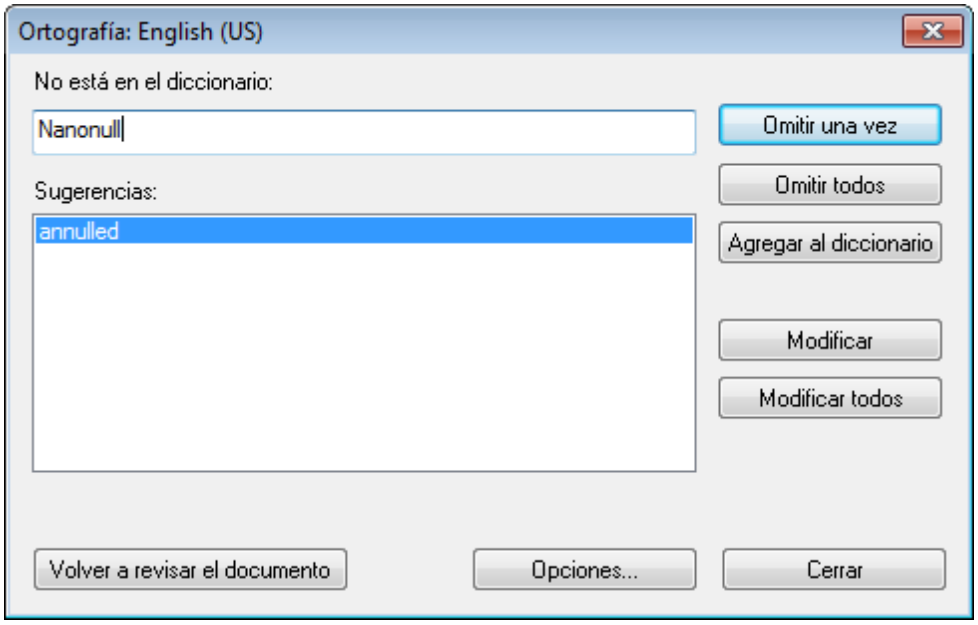


En este cuadro de diálogo puede seleccionar qué elementos y atributos se revisan y si las secciones CDATA y los comentarios se deben revisar o no. Consulte la descripción del comando **Opciones de ortografía** para más información.

Ejecutar la revisión ortográfica

El comando **Herramientas | Ortografía (Mayús+F7)** empieza a revisar el documento XML activo inmediatamente, según el contexto definido en el paso anterior. Si el corrector encuentra una palabra desconocida, aparece el cuadro de diálogo *Ortografía: No está en el diccionario* (*imagen siguiente)*. De lo contrario, el corrector recorre el documento hasta el final.

© 2015 Altova GmbH



A continuación describimos el cuadro de diálogo *Ortografía: No está en el diccionario* y las opciones disponibles en él.

No está en el diccionario
Este cuadro de texto contiene la palabra desconocida (es decir, una palabra que no está en el diccionario seleccionado ni en el diccionario del usuario). En caso de encontrar una palabra desconocida hay varias opciones:

- Puede editar la palabra en el cuadro de texto manualmente o seleccionar una de las opciones del panel *Sugerencias*. Después haga clic en el botón **Modificar** para reemplazar la palabra en el documento XML con la palabra nueva. (Si hace doble clic en una sugerencia, la palabra sugerida se inserta directamente en el documento XML.) La palabra que aparece en el cuadro *No está en el diccionario* también aparece resaltada en el documento XML para que pueda editarla directamente en el documento si lo prefiere. Si hace clic en el botón **Modificar todos** se reemplazan todas las instancias de la palabra en el documento con la palabra nueva.
- Puede decidir no realizar ningún cambio y omitir la advertencia del corrector (para la instancia actual de la palabra o para todas las instancias de la palabra).
- Puede añadir la palabra al diccionario del usuario para que a partir de ahora la palabra sea correcta.

Sugerencias
Este cuadro sugiere palabras que se parecen a la palabra desconocida (las palabras sugeridas provienen del diccionario integrado y de los diccionarios del usuario). Si hace doble clic en una palabra de la lista, la palabra elegida se inserta automáticamente en el documento y el proceso de revisión ortográfica continúa.

Omitir una vez
Este comando permite seguir revisando el documento e ignorar la primera aparición de la palabra desconocida. Si la misma palabra aparece otra vez en el documento, volverá a aparecer en el

recuadro *No está en el diccionario.*

Omitir todos
Este comando permite seguir revisando el documento e ignorar todas las apariciones de la palabra desconocida en todo el documento.

Agregar al diccionario
Este comando añade la palabra desconocida al diccionario del usuario. Para tener acceso al diccionario del usuario (para poder editarlo) utilice el cuadro de diálogo Opciones de ortografía.

Modificar
Este comando reemplaza la palabra desconocida en el documento XML con la palabra (nueva) del cuadro *No está en el diccionario*.

Modificar todos
Este comando reemplaza todas las instancias de la palabra desconocida en el documento XML con la palabra (nueva) del cuadro *No está en el diccionario*.

Volver a revisar el documento
Este comando vuelve a revisar el documento desde el principio.

Opciones
Dependiendo de la vista en la que esté trabajando, este botón abre:

- el cuadro de diálogo Opciones de ortografía (si está trabajando en la vista Authentic) o

Para más información sobre estos cuadros de diálogo consulte el apartado Opciones de ortografía.

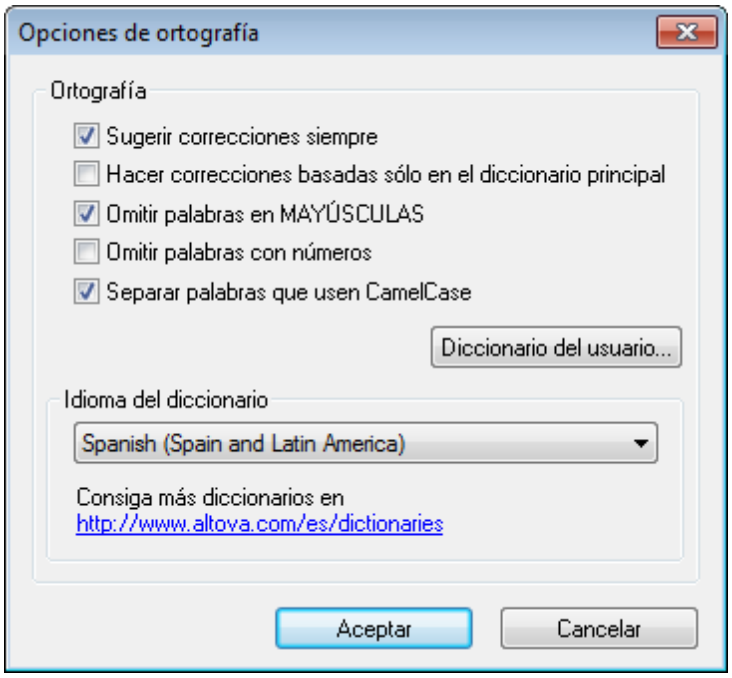
Cerrar
Este comando cierra el cuadro de diálogo Ortografía.

© 2015 Altova GmbH

10.8.2 Opciones de ortografía

El comando **Herramientas | Opciones de ortografía** abre las opciones de configuración del corrector ortográfico.

Opciones de ortografía
El cuadro de diálogo "Opciones de ortografía" (*imagen siguiente*) sirve para configurar el corrector ortográfico de forma general.



Sugerir correcciones siempre
Si marca esta casilla, el cuadro *Sugerencias* siempre mostrará opciones del diccionario integrado elegido y del diccionario del usuario. Si desactiva esta opción, el corrector no ofrecerá ninguna sugerencia.

Hacer correcciones basadas sólo en el diccionario principal
Si marca esta casilla, solamente se usan sugerencias del diccionario integrado elegido (diccionario principal). El diccionario del usuario no se utilizará para ofrecer sugerencias. Si marca esta casilla, se deshabilita el botón **Diccionario del usuario**. Es decir, el diccionario del usuario no se puede editar mientras esté activa esta opción.

Omitir palabras en MAYÚSCULAS
Si marca esta casilla, el corrector ortográfico pasa por alto las palabras que están en mayúsculas.

Omitir palabras con números
Si marca esta casilla, el corrector ortográfico pasa por alto las palabras que contienen números.

Separar palabras que usen CamelCase
Las palabras que usan CamelCase están formadas por palabras que empiezan por mayúsculas, unidas unas a otras sin espacios. Por ejemplo, la palabra *CamelCase* está formada por las

palabras *Camel* y *Case*, ambas en mayúsculas y unidas sin espacios. Este tipo de palabras no suele aparecer en los diccionarios y, por tanto, el corrector ortográfico las marcará como erróneas. Marque la casilla *Separar palabras que usen CamelCase* para que el corrector revise por separado cada una de las palabras que forman la palabra CamelCase. Esta opción está marcada por defecto.

Idioma del diccionario
Use este cuadro combinado para seleccionar el idioma del diccionario integrado del corrector ortográfico. La opción predeterminada es "US English" (inglés EE UU). Para descargar gratis más diccionarios en otros idiomas, visite el sitio web de Altova.

Agregar diccionarios para el corrector ortográfico
Por cada idioma hay dos archivos de diccionario Hunspell que funcionan conjuntamente: un archivo `.aff` y un archivo `.dic`. Los diccionarios se instalan en la carpeta `Lexicons` de este directorio:

Windows 7 y Windows 8:	`C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons`
Windows XP:	`C:\Documents and Settings\All Users\Datos de programa\Altova\SharedBetweenVersions\SpellChecker\Lexicons`

Dentro de la carpeta `Lexicons` se crea una carpeta por idioma: `<nombre del idioma>\<archivos del diccionario>`. Por ejemplo, en Windows 7 y Windows 8 los archivos de los dos diccionarios de español (de España y de Hispanoamérica y España) se almacenan así:

```
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\Spanish (Spain and Latin America)\es_ANY.aff
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\Spanish (Spain and Latin America)\es_ANY.dic
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\Spanish (Spain)\es_ES.dic
C:\ProgramData\Altova\SharedBetweenVersions\SpellChecker\Lexicons\Spanish (Spain)\es_ES.dic
```

La lista desplegable *Idioma del diccionario* del cuadro de diálogo "Opciones de ortografía" muestra los diccionarios disponibles. Se trata de los diccionarios de la carpeta `Lexicons` y reciben el nombre de las subcarpetas de la carpeta `Lexicons`. Por ejemplo, los dos diccionarios de inglés del ejemplo anterior aparecerían así en el cuadro combinado: *Spanish (Spain)* y *Spanish (Spain and Latin America)*.

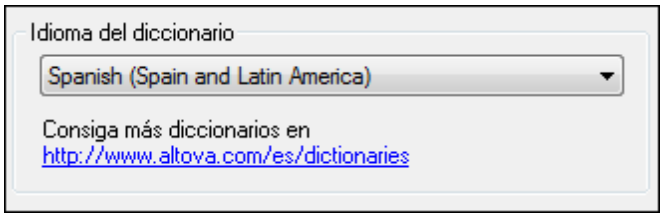
Todos los diccionarios instalados son compartidos por los diferentes usuarios del equipo y por las diferentes versiones de los productos de Altova (tanto en 64 como en 32 bits).

Hay dos maneras de agregar diccionarios nuevos para el corrector ortográfico. En ninguno de los dos casos es necesario registrar los archivos en el sistema:

- Puede añadir diccionarios Hunspell a una subcarpeta nueva de la carpeta `Lexicons`. Los diccionarios Hunspell se pueden descargar desde http://wiki.services.openoffice.org/wiki/Dictionaries o desde http://extensions.services.openoffice.org/en/dictionaries, por ejemplo. (Recuerde que OpenOffice utiliza el formato comprimido `OXT`. Cambie la extensión a `.zip` y descomprima los archivos `.aff` y `.dic` en las subcarpetas correspondientes de la carpeta `Lexicons`. También puede usar diccionarios Myspell, ya que los diccionarios Hunspell están basados en Myspell.)

© 2015 Altova GmbH

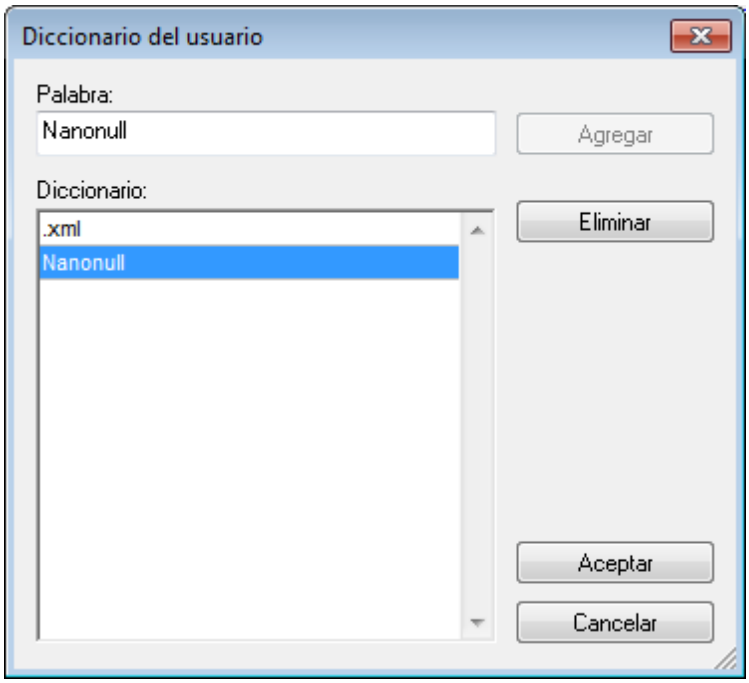
- Puede usar el instalador de diccionarios de Altova, que instala un paquete con varios diccionarios en el directorio adecuado del equipo. En el cuadro de diálogo "Opciones de ortografía", bajo el panel *Idioma del diccionario*, aparece un enlace a la página de Altova de descarga de diccionarios (*imagen siguiente*). Si no usa derechos de administrador para instalar los diccionarios, se producirá un error de instalación.



Nota: recuerde que es decisión suya aceptar o no las condiciones de uso de la licencia del diccionario elegido. También es responsabilidad suya comprobar si el diccionario puede utilizarse en su equipo o no.

Trabajar con el diccionario del usuario

Cada usuario tiene un diccionario del usuario propio, donde se almacenan las palabras aprobadas por el usuario. Durante la revisión ortográfica, el corrector compara la ortografía con una lista compuesta por palabras del diccionario integrado y del diccionario del usuario. Puede añadir o eliminar palabras del diccionario del usuario en el cuadro de diálogo "Diccionario del usuario" (*imagen siguiente*) Para abrir este cuadro de diálogo pulse el botón **Diccionario del usuario** del cuadro de diálogo "Opciones de ortografía" (*segunda imagen de este apartado*).



Para añadir una palabra al diccionario del usuario escriba la palabra en el recuadro *Palabra* y pulse el botón **Agregar**. La palabra se añade a la lista alfabética del panel *Diccionario.* Para eliminar una palabra del diccionario, seleccione la palabra del panel *Diccionario* y pulse el botón **Eliminar**. La palabra se elimina del panel *Diccionario*. Cuando termine de editar el cuadro de diálogo "Diccionario del usuario" haga clic en el botón **Aceptar**. Los cambios se guardan en el diccionario del usuario.

También puede añadir palabras al diccionario del usuario durante la revisión ortográfica. Si el corrector encuentra una palabra desconocida, aparece el cuadro de diálogo Ortografía. Pulse el botón **Agregar al diccionario** para añadir la palabra desconocida al diccionario del usuario.

El diccionario del usuario se encuentra en este directorio:

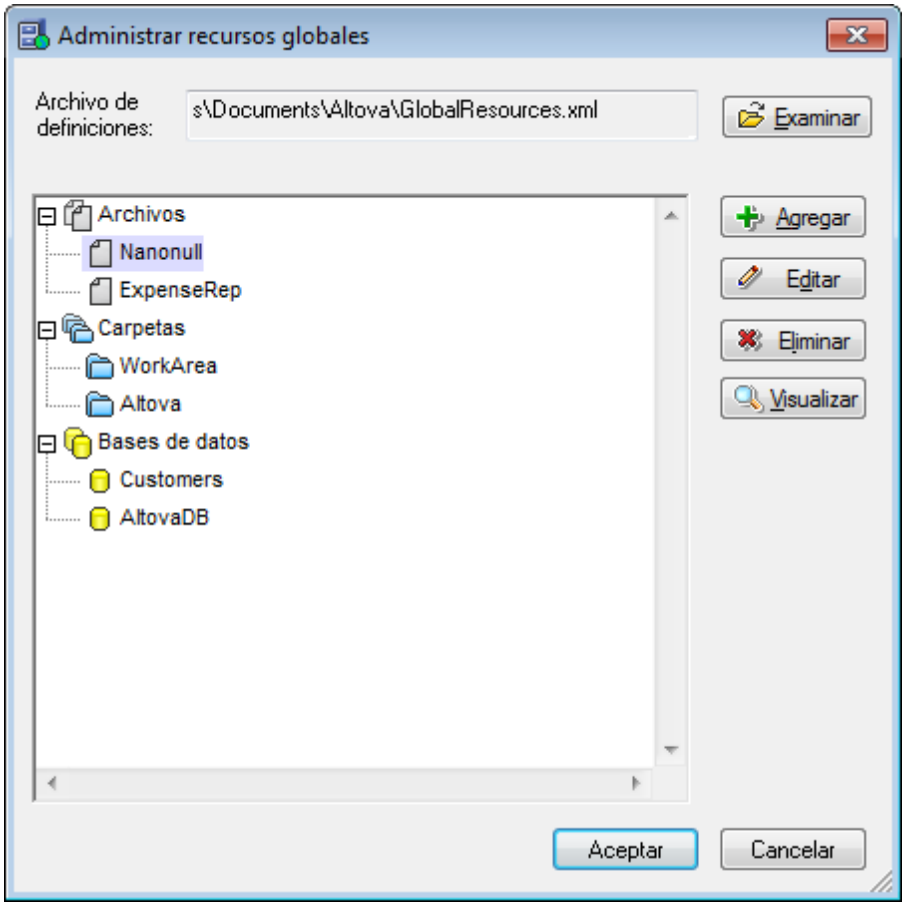
Windows 7 y Windows 8: `C:\Users\<usuario>\Mis Documentos\Altova\SpellChecker\Lexicons\user.dic`
Windows XP: `C:\Documents and Settings\<usuario>\Mis Documentos\Altova\SpellChecker\Lexicons\user.dic`

© 2015 Altova GmbH

10.8.3 Recursos globales

El comando **Recursos globales** abre el cuadro de diálogo "Administrar recursos globales" (*imagen siguiente*), donde puede:

- Seleccionar qué archivo XML de recursos globales se utiliza para los recursos globales.
- Añadir recursos globales (o alias) de tipo archivo, carpeta o base de datos.
- Especificar varias configuraciones para cada recurso global (alias). Cada configuración se asigna a un recurso concreto.

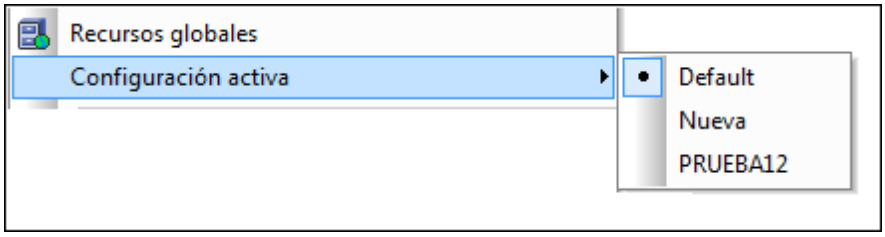


Para más información consulte la sección Definir recursos globales.

Nota: también puede abrir el cuadro de diálogo "Administrar recursos globales" desde la barra de herramientas Recursos globales (**Herramientas | Personalizar | Barras de herramientas | Recursos globales**).

10.8.4 Configuración activa

Al pasar el puntero del mouse sobre el comando **Configuración activa** aparece un submenú con todas las configuraciones definidas en el archivo XML de recursos globales activo (*imagen siguiente*).



La configuración que está activa actualmente se identifica con un punto. En la imagen anterior, por ejemplo, la configuración activa es `Default`. Para cambiar de configuración simplemente haga clic en otra configuración.

Nota: también puede cambiar de configuración en la barra de herramientas Recursos globales (**Herramientas | Personalizar | Barras de herramientas | Recursos globales**).

© 2015 Altova GmbH

10.8.5 Personalizar

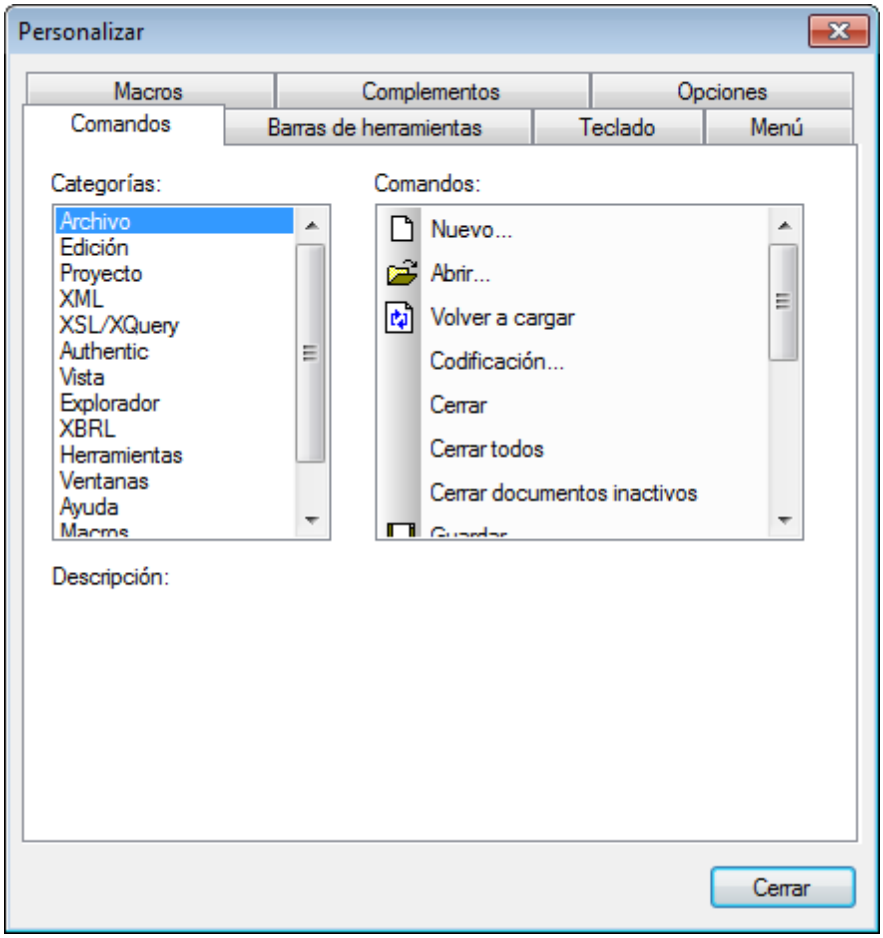
El comando **Personalizar** sirve para adaptar los menús y barras de herramientas de la aplicación a sus requisitos. Al hacer clic en este comando aparece el cuadro de diálogo "Personalizar", que está compuesto por estas pestañas:

- *Comandos*: desde esta pestaña puede arrastrar comandos de la aplicación y de macros hasta barras de menú, menús y barras de herramientas.
- *Barras de herramientas*: aquí puede activar, desactivar y restaurar cada una de las barras de herramientas.
- *Teclado*: aquí puede crear teclas de acceso directo para cada comando de la aplicación y de macros.
- *Menú*: aquí puede seleccionar y activar las barras de menú y los menús contextuales que desea personalizar. Esta pestaña funciona conjuntamente con la pestaña *Comandos*.
- Complementos: aquí puede activar complementos e integrarlos a la interfaz.
- Opciones: aquí puede configurar la presentación de las barras de herramientas.

En este apartado también describimos el menú contextual que aparece si hace clic con el botón secundario en una barra de menú, en un menú o en una barra de herramientas de la aplicación mientras está abierto el cuadro de diálogo "Personalizar".

Comandos

En la pestaña ***Comandos*** puede personalizar sus menús y barras de herramientas y añadir comandos a los menús y a las barras de herramientas, dependiendo de lo que necesite. No obstante, tenga en cuenta que no puede crear comandos ni menús nuevos para la aplicación.



Para añadir un comando a una barra de herramientas o menú:

1. Seleccione el comando **Herramientas | Personalizar**. Se abre el cuadro de diálogo "Personalizar".
2. Seleccione la pestaña ***Comandos***. En el cuadro de lista *Categorías* seleccione la opción **Todos los comandos**. Todos los comandos disponibles aparecen en el cuadro de lista *Comandos*.
3. Haga clic en un comando del cuadro de lista *Comandos* y arrástrelo a un menú o barra de herramientas ya existente. Al pasar el puntero por encima de una posición donde se puede colocar el comando aparece el icono **I**.
4. Cuando encuentre la posición donde desea colocar el comando, suelte el botón del mouse.

Tenga en cuenta que:

- Mientras arrastra el comando, aparece un pequeño botón al final del puntero del mouse. Esto indica que el comando está siendo arrastrado.
- Si el comando no se puede colocar en la posición actual del cursor, debajo del puntero aparece una **X**.
- Si el cursor está en una posición donde se puede colocar el comando (en una barra de herramientas o en un menú), la **X** desaparece y el icono **I** indica que la posición es válida.
- Los comandos se pueden colocar en menús o barras de herramientas. Si creó una barra de herramientas nueva, puede usar este mecanismo de personalización para rellenar la

 © 2015 Altova GmbH

barra de herramientas con comandos.
- Si pasa el cursor por un menú que está cerrado, el menú se abre y puede insertar el comando en cualquier parte del menú.

Agregar comandos a menús contextuales

También puede añadir comandos a menús contextuales arrastrando comandos del cuadro de lista *Comandos* hasta el menú contextual:

1. Haga clic en la pestaña *Menú* del cuadro de diálogo Personalizar.
2. En el cuadro combinado del panel *Menús contextuales* seleccione un menú contextual. El menú contextual seleccionado aparece en pantalla.
3. Vuelva a la pestaña *Comandos* del cuadro de diálogo Personalizar.
4. Seleccione un comando en el cuadro de lista *Comandos* y arrástrelo hasta la posición deseada del menú contextual.

Eliminar un comando o menú

Por último, puede eliminar un comando de un menú, menú contextual (ver párrafo anterior) o barra de herramientas o eliminar un menú entero:

1. Abra el cuadro de diálogo "Personalizar" (**Herramientas | Personalizar**).
2. Seleccione cualquier pestaña del cuadro de diálogo "Personalizar". Haga clic con el botón secundario en un menú o comando de menú y seleccione **Eliminar** en el menú contextual que aparece. Si lo prefiere, también puede arrastrar el menú o comando de menú hasta que aparezca el icono **X** debajo del puntero del mouse y suelte el menú o comando de menú. Como resultado se elimina el menú o comando de menú.

Para volver a instalar los comandos de menú eliminados, utilice los mecanismos descritos en este apartado. Para restablecer un menú eliminado, seleccione **Herramientas | Personalizar | Menú** y pulse el botón **Restaurar** del panel *Menús del marco de la aplicación*. Otra opción es seleccionar **Herramientas | Personalizar | Barras de herramientas**, hacer clic en la barra de herramientas pertinente y pulsar el botón **Restaurar**.

Barras de herramientas

En la pestaña ***Barras de herramientas*** puede: (i) activar o desactivar barras de herramientas (es decir, decidir qué barras de herramientas aparecen en la interfaz), (ii) definir qué iconos aparecen en cada barra de herramientas y (iii) crear barras de herramientas personalizadas.

Las barras de herramientas incluyen iconos para los comandos de menú más utilizados. Además, al pasar el puntero sobre un icono, se ofrece información rápida sobre el icono en un mensaje emergente y en la barra de estado de la aplicación. Las barras de herramientas se pueden colocar en cualquier posición de la pantalla, donde aparece como ventana flotante.

Nota: para añadir un comando a una barra de herramientas, arrastre un comando del cuadro de lista *Comandos* de la pestaña ***Comandos*** hasta la barra de herramientas. Para eliminar un comando de una barra de herramientas, abra el cuadro de diálogo Personalizar, seleccione cualquier pestaña y arrastre el comando fuera de la barra de herramientas (para más información consulte el apartado Comandos).

Nota: la configuración de una barra de herramientas definida en una vista determinada afecta solamente a dicha vista. Para que la configuración afecte a todas las vistas, marque la casilla situada al final de la pestaña *Barras de herramientas* del cuadro de diálogo Personalizar.

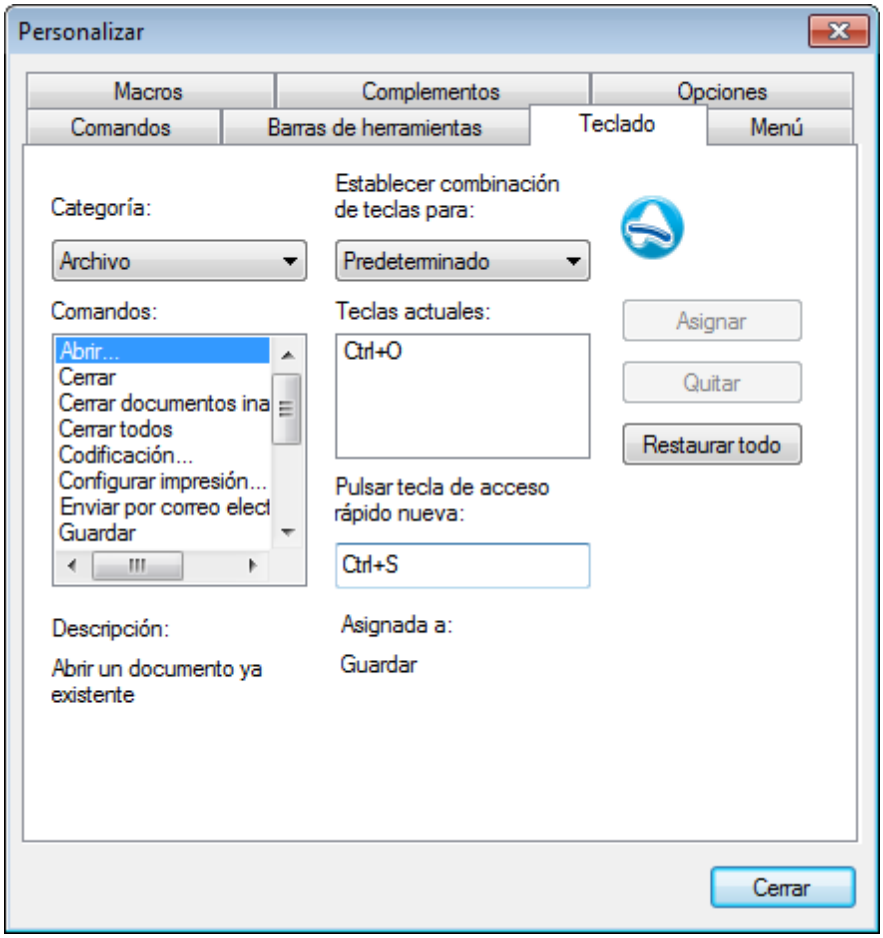
En esta pestaña puede llevar a cabo estas operaciones:

- *Para activar/desactivar una barra de herramientas:* marque su casilla en el cuadro de lista *Barras de herramientas*.
- *Para aplicar los cambios a todas las vistas:* marque la casilla situada al final de la pestaña. De lo contrario, los cambios realizados afectan solamente a la vista activa. Recuerde que los cambios realizados **después** de marcar la casilla *Aplicar cambios en todas las vistas* afectarán a todas las vistas.
- *Para añadir una barra de herramientas nueva:* pulse el botón **Nueva...** y escriba el nombre de la barra de herramientas nuevas en el cuadro de diálogo "Nombre de la barra de herramientas" que aparece. Arrastre comandos desde la pestaña *Comandos* hasta la barra de herramientas nueva.
- *Para cambiar el nombre de una barra de herramientas nueva:* seleccione la barra de herramientas en el panel *Barra de herramientas* y pulse el botón **Cambiar de nombre**. Edite el nombre en el cuadro de diálogo "Nombre de la barra de herramientas" que aparece.
- *Para restaurar la barra de menús:* seleccione `Barra de menús` en el panel *Barras de herramientas* y pulse el botón **Restaurar**. La barra de menús vuelve a su estado original de instalación.
- *Para restaurar todas las barras de herramientas y comandos de menú:* pulse el botón **Restaurar todo**. Todas las barras de herramientas y menús vuelven a su estado original de instalación.
- *Para eliminar una barra de herramientas:* seleccione la barra de herramientas en el panel *Barras de herramientas* y pulse el botón **Eliminar**.
- *Para mostrar las etiquetas de texto de una barra de herramientas:* seleccione la barra de herramientas y marque la casilla *Mostrar etiquetas de texto.* Recuerde que debe activar las etiquetas de texto de cada barra de herramientas por separado.

Teclado

En la pestaña ***Teclado*** puede crear teclas de acceso rápido nuevas o cambiar las teclas de acceso rápido ya existentes para cualquier comando de la aplicación.

© 2015 Altova GmbH



Siga estos pasos para asignar una tecla de acceso rápido nueva a un comando o cambiar una tecla de acceso rápido ya existente:

1. En el cuadro combinado *Categoría* seleccione la opción *Todos los comandos*.
2. En el cuadro de lista *Comandos* seleccione el comando al que desea asignar una tecla de acceso rápido nueva o el comando cuya tecla de acceso rápido desea cambiar.
3. Haga clic dentro del cuadro *Pulsar tecla de acceso rápido nueva* y pulse la tecla de acceso rápido que desea asignar al comando. La tecla de acceso rápido aparece en el cuadro *Pulsar tecla de acceso rápido nueva.* Si la tecla de acceso rápido no se asignó todavía a ningún comando, se habilita el botón **Asignar**. Si la tecla ya se asignó a un comando, el comando aparece debajo del cuadro y el botón **Asignar** está deshabilitado. (Para borrar el contenido del cuadro *Pulsar tecla de acceso rápido nueva* pulse **Ctrl**, **Alt** o **Mayús**).
4. Haga clic en el botón **Asignar**. La tecla de acceso rápido aparece ahora en el cuadro de lista *Teclas actuales*. Puede asignar varias teclas de acceso rápido al mismo comando si lo desea.
5. Para confirmar los cambios pulse el botón **Cerrar**.

Eliminar una tecla de acceso rápido.

No se puede asignar la misma tecla de acceso rápido a comandos diferentes. Si desea eliminar una tecla de acceso rápido, selecciónela en el cuadro *Teclas actuales* y después pulse el botón

Quitar. Después pulse el botón **Cerrar**.

Establecer combinación de teclas para
Actualmente no tiene ninguna función

© 2015 Altova GmbH

Teclas de acceso rápido predeterminadas (ordenadas por tecla)

La tabla que aparece a continuación enumera las teclas de acceso rápido asignadas a los comandos más utilizados en el momento de la instalación.

Tecla de acceso rápido	Comando
F1	Menú Ayuda
F3	Buscar siguiente
F5	Actualizar
F7	Comprobar formato XML
F8	Validar el documento XML
F10	Transformación XSL
Ctrl+F10	Transformación XSL:FO
F11	Paso a paso por instrucciones
Ctrl+F11	Paso a paso por procedimientos
Mayús + F11	Paso a paso para salir
Núm +	Expandir
Núm -	Contraer
Núm *	Expandir todos
Ctrl+Núm-	Contraer elementos no seleccionados
Ctrl + G	Ir a la línea o carácter
Ctrl+Tabulador	Alterna entre dos documentos
Ctrl+F6	Recorre las ventanas abiertas
Flechas de dirección	
(arriba / abajo)	Mueve la barra de selección
Esc.	Abandona los cambios o cierra el cuadro de diálogo
Tecla Entrar / Barra espaciadora	Confirma una selección
`Alt + F4`	Cierra Authentic Desktop
Ctrl + F4	Cierra la ventana activa
Alt + F, 1	Abre el último archivo

Ctrl + N	Archivo nuevo
Ctrl + O	Abrir archivo
Ctrl + S	Guardar archivo
Ctrl + P	Imprimir archivo
Ctrl + A	Seleccionar todo
Mayús + Supr	Cortar (o Ctrl + X)
Ctrl + C	Copiar
Ctrl + V	Pegar
Ctrl + Z	Deshacer
Ctrl + Y	Rehacer
Supr	Eliminar
Ctrl + F	Buscar
F3	Buscar siguiente
Ctrl + H	Reemplazar
Ctrl + I	Anexar atributo
Ctrl + E	En la vista Cuadrícula, Anexar elemento. En la vista Texto, ir a la etiqueta inicial/final cuando el cursor está en el otro miembro del par
Ctrl + T	Anexar texto
Ctrl + D	Anexar sección CDATA
Ctrl + M	Anexar comentario
Ctrl + Mayús + I	Insertar atributo
Ctrl + Mayús + E	Insertar elemento
Ctrl + Mayús + T	Insertar texto
Ctrl + Mayús + D	Insertar sección CDATA
Ctrl + Mayús + M	Insertar comentario
Ctrl + Alt + I	Agregar atributo secundario
Ctrl + Alt + E	Agregar elemento secundario
Ctrl + Alt + T	Agregar texto secundario
Ctrl + Alt + D	Agregar sección CDATA secundaria
Ctrl + Alt + M	Agregar comentario secundario

Teclas de acceso rápido de la vista Texto	
Ctrl + "+"	Acercarse
Ctrl + "-"	Alejarse

© 2015 Altova GmbH

Ctrl + 0	Restaurar el nivel de zoom
Ctrl + rueda del mouse hacia adelante	Acercarse
Ctrl + rueda del mouse hacia atrás	Alejarse

Teclas de acceso rápido predeterminadas (ordenadas por función)

La tabla que aparece a continuación enumera las teclas de acceso rápido asignadas a los comandos más utilizados en el momento de la instalación, ordenadas alfabéticamente por función.

Función/Comando	Tecla de acceso rápido
Abandonar los cambios	Esc.
Abrir archivo	Ctrl + O
Abrir el último archivo	Alt + F, 1
Actualizar	F5
Agregar atributo secundario	Ctrl + Alt + I
Agregar comentario secundario	Ctrl + Alt + M
Agregar elemento secundario	Ctrl + Alt + E
Agregar sección CDATA secundaria	Ctrl + Alt + D
Agregar texto secundario	Ctrl + Alt + T
Anexar atributo	Ctrl + I
Anexar comentario	Ctrl + M
Anexar elemento	Ctrl + E (Vista Cuadrícula)
Anexar sección CDATA	Ctrl + D
Anexar texto	Ctrl + T
Archivo nuevo	Ctrl + N
Buscar	Ctrl + F
Buscar siguiente	F3
Cerrar Authentic Desktop	Alt + F4
Cerrar la ventana activa	Ctrl + F4
Comprobar formato XML	F7
Confirma una selección	Tecla Entrar / Barra espaciadora
Contraer	Núm -
Contraer elementos no seleccionados	Ctrl + Núm-
Copiar	Ctrl + C
Cortar	Mayús + Supr (o Ctrl + X)
Deshacer	Ctrl + Z
Eliminar elemento	Supr
Expandir	Núm +
Expandir todo	Núm *
Guardar archivo	Ctrl + S
Imprimir archivo	Ctrl + P
Iniciar el depurador	Alt + F11

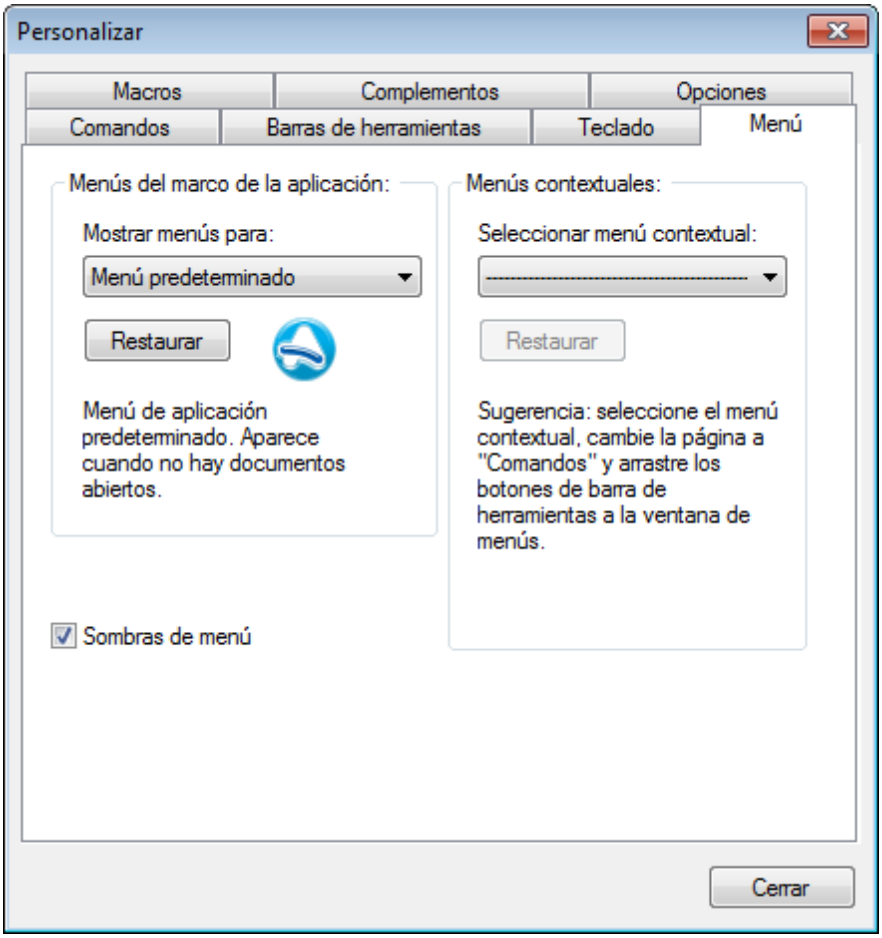
© 2015 Altova GmbH

Insertar atributo	Ctrl + Mayús + I
Insertar comentario	Ctrl + Mayús + M
Insertar elemento	Ctrl + Mayús + E
Insertar sección CDATA	Ctrl + Mayús + D
Insertar texto	Ctrl + Mayús + T
Ir a la etiqueta inicial/final cuando el cursor está en el otro miembro del par	Ctrl + E (Vista Texto)
Ir a la línea o carácter	Ctrl + G
Menú Ayuda	F1
Mover la barra de selección	Teclas de dirección (arriba / abajo)
Paso a paso para salir	Mayús + F11
Paso a paso por instrucciones	F11
Paso a paso por procedimientos	Ctrl + F11
Pegar	Ctrl + V
Recorrer las ventanas abiertas	Ctrl + Tabulador y Ctrl + F6
Reemplazar	Ctrl + H
Rehacer	Ctrl + Y
Seleccionar todo	Ctrl + A
Transformación XSL	F10
Transformación XSL:FO	Ctrl + F10
Validar el documento XML	F8
Ver la definición de un elemento	Ctrl + doble clic en un elemento

En la aplicación puede ver una lista de los comandos, su descripción y sus teclas de acceso directo en el cuadro de diálogo "Mapa de teclado" (**Ayuda | Mapa de teclado**).

Menú

En la pestaña ***Menú*** puede personalizar las dos barras de menú principales (la barra de menú predeterminada y la barra de menú de la aplicación) así como los menús contextuales de la aplicación.



Personalizar la barra de menú predeterminada y la barra de menú de la aplicación

La barra de menú predeterminada es la barra de menú que aparece cuando no hay ningún documento abierto en la ventana principal. La barra de menú de la aplicación es la barra que aparece cuando hay un documento abierto en la ventana principal. Cada una de estas barras de menú se puede personalizar y los cambios realizados en una de las barras de menú no afecta a la otra.

Para personalizar una barra de menú, selecciónela en el cuadro combinado *Mostrar menús para:* de la pestaña *Menú* (*imagen anterior*). Después cambie a la pestaña *Comandos* del cuadro de diálogo Personalizar y arrastre comandos desde el cuadro de lista *Comandos* hasta la barra de menú.

Eliminar comandos de menús y restaurar las barras de menú

Para eliminar un menú entero o un comando de un menú, seleccione el menú o el comando de menú y después (i) haga clic con el botón secundario y seleccione **Eliminar** o (ii) arrastre el comando fuera de la barra de menú o del menú.

Para restaurar estas dos barras de menú (la barra de menú predeterminada y la de la aplicación) a su estado original de instalación seleccione el menú en el cuadro combinado *Mostrar menús para:* y haga clic en el botón **Restaurar** situado bajo el cuadro combinado.

© 2015 Altova GmbH

Personalizar los menús contextuales de la aplicación

Los menús contextuales son los menús que aparecen cuando se hace clic con el botón secundario en determinados objetos de la interfaz de la aplicación. Siga estos pasos para personalizar un menú contextual:

1. Seleccione el menú contextual en el cuadro combinado *Seleccionar menú contextual*. Aparece el menú contextual.
2. Pase a la pestaña *Comandos* del cuadro de diálogo Personalizar.
3. Arrastre un comando del cuadro de lista *Comandos* al menú contextual.
4. Si desea eliminar un comando del menú contextual, haga clic en él con el botón secundario y seleccione **Eliminar**. También puede seleccionar el comando y arrastrarlo fuera del menú contextual.

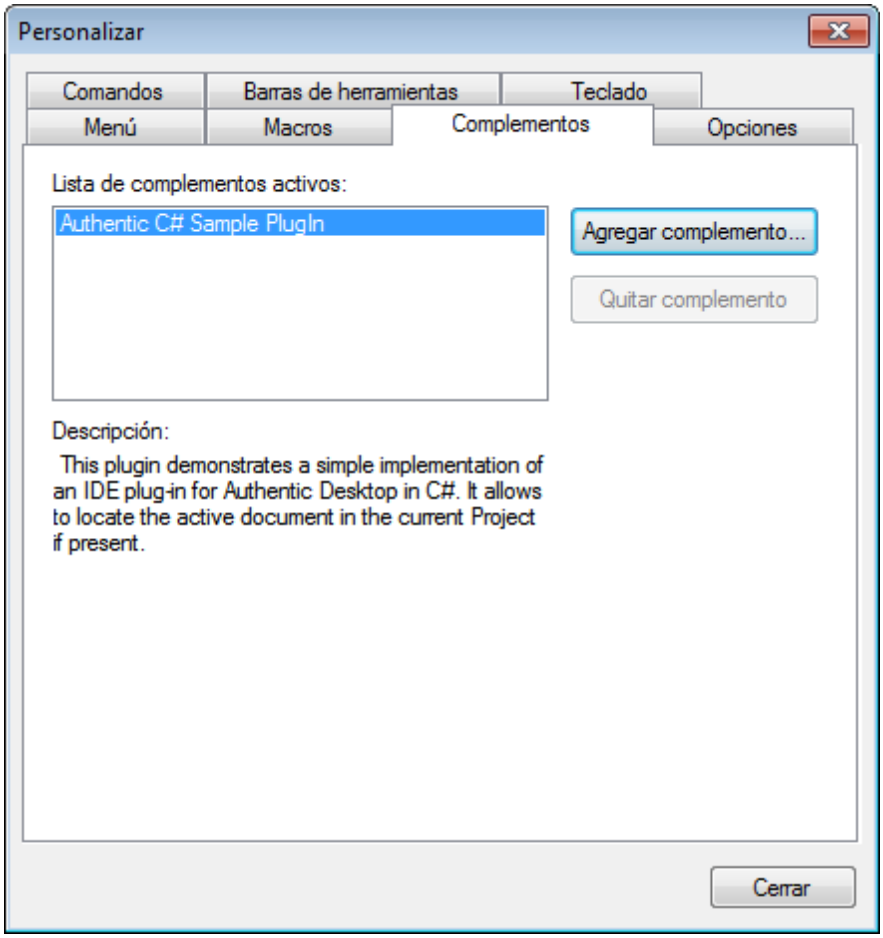
Para restaurar un menú contextual a su estado original de instalación seleccione el menú en el cuadro combinado *Seleccionar menú contextual* y después pulse el botón **Restaurar**, situado bajo el cuadro combinado.

Sombras de menú

Marque la casilla *Sombras de menú* para dar sombra a todos los menús.

Complementos

Desde la pestaña ***Complementos*** puede integrar complementos y colocar comandos (si se han programado) en un menú o en una barra de herramientas de la aplicación. Pulse el botón **Agregar complemento** de la pestaña *Complementos* (*imagen siguiente*) y busque el archivo DLL del complemento (consulte más abajo *'Crear complemento').* Haga clic en **Aceptar** para agregar el complemento. Si quiere, puede agregar varios complementos.



En el cuadro de diálogo aparece una descripción de los complementos añadidos y se habilita el botón **Quitar complemento**. Si el código del complemento crea barras de herramientas y menús, estos aparecen inmediatamente en la interfaz de la aplicación. Si desea quitar un complemento, selecciónelo y pulse el botón **Quitar complemento**.

Cómo crear complementos

En la carpeta `Examples\IDEPlugin` de la aplicación encontrará el código fuente de complementos de muestra. Para generar un complemento desde por ejemplo un nodo fuente, siga los siguientes pasos:

1. Abra la solución con la que desea generar el complemento en Visual Studio.
2. Genere el complemento con el comando del menú Crear.
3. El DLL del complemento se creará en la carpeta Bin o Debug. Este archivo DLL es el archivo que se debe de agregar como complemento (véase arriba).

Para más información sobre complementos consulte la sección Complementos para entornos IDE.

© 2015 Altova GmbH

Opciones

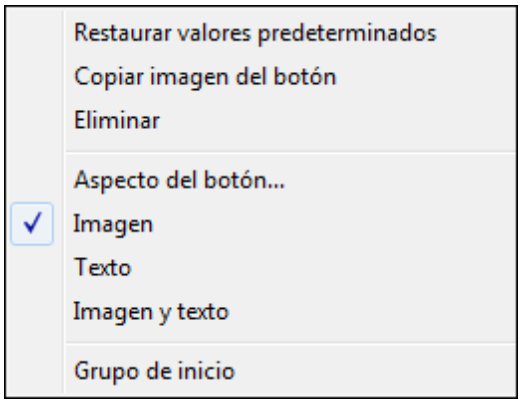
En la pestaña ***Opciones*** puede definir la configuración general del entorno.

Marque las casillas para activar estas opciones:

- *Mostrar información en pantalla en las barras de herramientas:* al pasar el puntero del mouse sobre los iconos de las barras de herramientas aparece un mensaje emergente con una breve descripción de la función del icono, así como su tecla de acceso rápido (si le asignó una y si marcó la casilla *Mostrar teclas de acceso rápido*).
- *Mostrar teclas de acceso rápido en la información en pantalla:* la información en pantalla de los iconos de las barras de herramientas incluye la tecla de acceso rápido del comando.
- *Iconos grandes*: alterna el tamaño de los iconos de las barras de herramientas entre tamaño estándar y tamaño grande.

Menú contextual Personalizar

El menú contextual **Personalizar** (*imagen siguiente*) es el menú que aparece al hacer clic con el botón secundario en un menú, un comando de menú o en un icono de barra de herramientas de la aplicación mientras está abierto el cuadro de diálogo Personalizar.

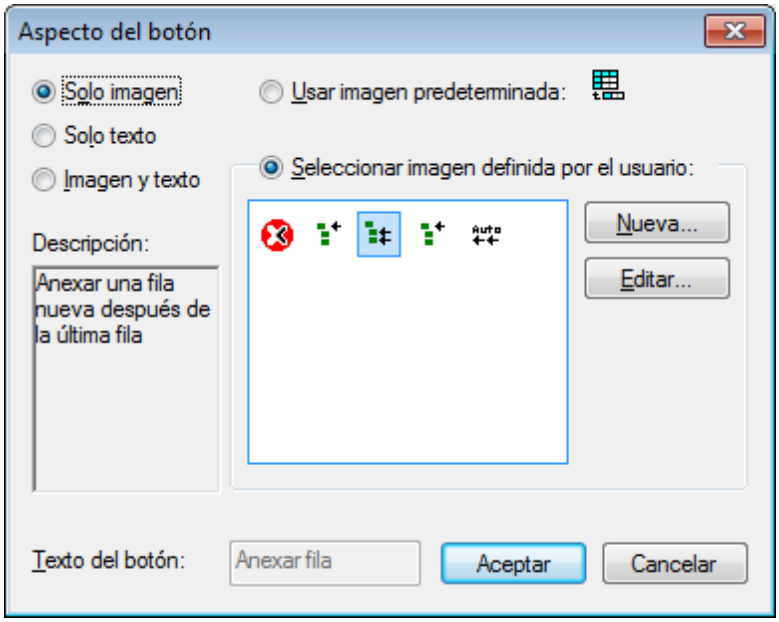


Estas son las opciones del menú contextual **Personalizar**:

- *Restaurar valores predeterminados* (por ahora no tiene función alguna).
- *Copiar imagen del botón:* copia la imagen del icono en el portapapeles.
- *Eliminar:* elimina el menú, comando o icono seleccionado. Para más información sobre cómo restaurar los componentes eliminados lea el último párrafo de este apartado.
- *Aspecto del botón:* abre el cuadro de diálogo "Aspecto del botón" (*imagen siguiente*), donde puede configurar las propiedades que definen el aspecto del icono seleccionado. Consulte el resto de este apartado para más información.
- *Imagen, Texto, Imagen y texto:* estas tres opciones son mutuamente excluyentes y determinan si el icono seleccionado debe ser solamente icono, solamente texto o ambos. Esto también se puede definir en el cuadro de diálogo "Aspecto del botón".
- *Grupo de inicio:* inserta un divisor de grupo vertical a la izquierda del icono seleccionado. Como resultado el icono seleccionado se convierte en el primer icono del grupo.

El cuadro de diálogo "Aspecto del botón"

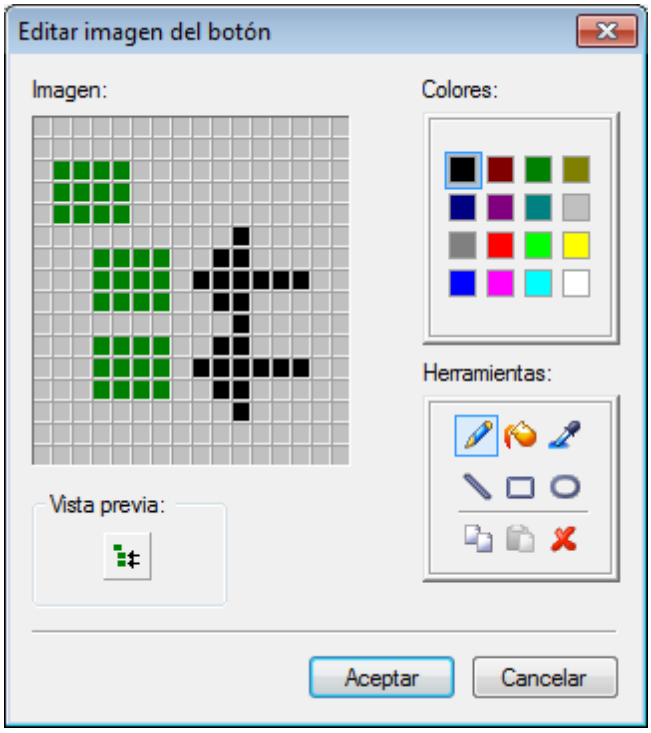
Si hace clic con el botón secundario en un icono de una barra de herramientas (mientras está abierto el cuadro de diálogo Personalizar) y después selecciona **Aspecto del botón**, aparece el cuadro de diálogo "Aspecto del botón" (*imagen siguiente*). En este cuadro de diálogo puede editar la imagen del icono y su texto. Actualmente sólo puede editar iconos de macros y complementos en este cuadro de diálogo.



Estas son las funciones de edición disponibles en este cuadro de diálogo para el icono seleccionado:

- *Solo imagen, Solo texto* o *Imagen y texto:* seleccione uno de estos tres botones de opción para definir el formato del icono de la barra de herramientas.
- *Edición de la imagen:* si selecciona la opción *Solo imagen* o *Imagen y texto*, se habilitan las opciones de edición de la imagen. Pulse el botón **Nueva** para crear una imagen nueva, que se añadirá al panel de imágenes definidas por el usuario. Después seleccione una imagen y pulse **Editar**.

© 2015 Altova GmbH



- *Selección de la imagen:* seleccione una imagen del panel de imágenes definidas por el usuario y después pulse **Aceptar** para usar la imagen seleccionada como icono de la barra de herramientas.
- *Edición y selección del texto:* si selecciona la opción *Solo texto* o *Imagen y texto*, se habilita el cuadro de texto *Texto del botón.* Escriba el texto y haga clic en **Aceptar** para usar este texto con el icono de la barra de herramientas.

Nota: el cuadro de diálogo "Aspecto del botón" también se puede usar para editar el texto de los comandos de menú. Haga clic con el botón secundario en el comando de menú (cuando está abierto el cuadro de diálogo Personalizar), seleccione **Aspecto del botón** y edite el texto del comando de menú en el cuadro de texto *Texto del botón*.

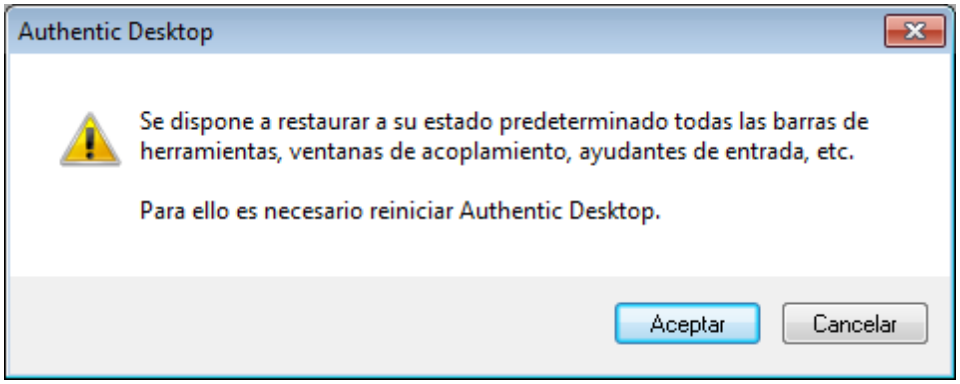
Restaurar menús, comandos e iconos de barras de herramientas eliminados

Si elimina un menú, comando o icono de barra de herramientas con el comando **Eliminar** del menú contextual Personalizar, puede restaurarlos siguiendo estos pasos:

- *Menús:* seleccione **Herramientas | Personalizar | Menú** y haga clic en el botón **Restaurar** del panel *Menús del marco de la aplicación.* Otra opción es seleccionar **Herramientas | Personalizar | Barras de herramientas**, marcar la casilla `Barra de menús` y pulsar el botón **Restaurar**.
- *Comandos de menú:* seleccione **Herramientas | Personalizar | Comandos** y arrastre el comando desde el cuadro de lista *Comandos* hasta el menú.
- *Iconos de barra de herramientas:* seleccione **Herramientas | Personalizar | Comandos** y arrastre el comando desde el cuadro de lista *Comandos* hasta la barra de herramientas.

10.8.6 Restaurar barras de herramientas y ventanas

El comando **Restaurar barras de herramientas y ventanas** cierra Authentic Desktop y lo reinicia con su configuración predeterminada. Antes de cerrarse, Authentic Desktop le pregunta si desea cerrar o no la aplicación (*imagen siguiente*).


Authentic Desktop
Se dispone a restaurar a su estado predeterminado todas las barras de herramientas, ventanas de acoplamiento, ayudantes de entrada, etc.
Para ello es necesario reiniciar Authentic Desktop.
Aceptar
Cancelar

Este comando es muy práctico si movió ventanas o barras de herramientas de sitio, si las ocultó o si ajustó su tamaño y desea poner todas estas barras de herramientas y ventanas como estaban en un principio.

© 2015 Altova GmbH

10.8.7 Opciones

El comando **Herramientas** | **Opciones** sirve para definir la configuración global de la aplicación. Estas opciones de configuración se definen en un cuadro de diálogo dividido en pestañas y se guardan en el registro. Las opciones elegidas en este cuadro de diálogo afectan a todas las ventanas de documento, presentes y futuras. El botón **Aplicar** del cuadro de diálogo Opciones muestra los cambios en los documentos abiertos y fija la configuración actual. Los cambios se pueden ver inmediatamente en las ventanas de fondo.

En esta sección describimos detalladamente todas las pestañas del cuadro de diálogo "Opciones".

Archivo

La pestaña ***Archivo*** define cómo se abren y guardan los documentos en Authentic Desktop. La pestaña Codificación incluye opciones de configuración relacionadas.

Abrir o crear archivo nuevo en la vista Cuadrícula

Puede elegir si los archivos se abren o se crean en la vista Cuadrícula o Texto. Si elige la vista Cuadrícula, también puede elegir si se expanden todas las líneas automáticamente.

Recarga automática de archivos cambiados

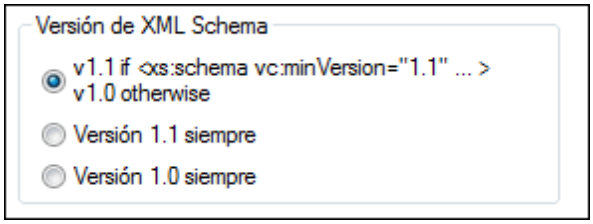
Si está trabajando en un entorno de varios usuarios o si trabaja con archivos que se generan de forma dinámica en un servidor, puede vigilar si se realizan cambios en los archivos que están abiertos en la interfaz. Cada vez que Authentic Desktop detecte un cambio en un documento de los que están abiertos, la aplicación le pregunta si desea volver a cargar el archivo en el que se realizaron los cambios.

Validación

Si usa archivos DTD o esquemas para definir la estructura de sus documentos XML, puede comprobar automáticamente la validez del documento cada vez que lo abra o lo guarde. También puede limitar la operación de validación a los archivos cuyo tamaño no supere un número determinado de MB. Si el documento no es válido, aparece un mensaje de error. Si es válido, no aparece ningún mensaje y la operación continúa sin emitir notificación alguna. Authentic Desktop también puede almacenar estos archivos en memoria para no tener que volver a cargarlos cada vez que los necesite (por ejemplo, cuando se accede a través de una URL al esquema al que se hace referencia). Si la declaración de ubicación del esquema usa una URL, deshabilite la opción *Almacenar archivos DTD o de esquema en memoria* para que los cambios realizados en el esquema aparezcan inmediatamente, en lugar de usar la versión en memoria del esquema.

Versión de XML Schema

El modo XSD que se habilita en la vista **Esquema** depende tanto de (i) la presencia/ausencia del atributo `/xs:schema/vc:minVersion` del documento XSD (y, si está presente, de su valor), como de (ii) la opción seleccionada en la pestaña *Archivo* del cuadro de diálogo Opciones (**Herramientas | Opciones**) (*imagen siguiente*).



En definitiva pueden darse todas las situaciones que aparecen en la siguiente tabla. Con *Versión de XML Schema* nos referimos a la opción seleccionada en el panel de opciones *Versión de XML Schema* (*imagen anterior*). La segunda columna (atributo `vc:minVersion`) corresponde al valor del atributo `xs:schema/@vc:minVersion` del esquema XML.

Versión de XML Schema	Atributo `vc:minVersion`	Modo XSD
Versión 1.0 siempre	Ausente (o presente con cualquier valor)	1.0
Versión 1.1 siempre	Ausente (o presente con cualquier valor)	1.1
Valor de @vc:minVersion	`1.1`	1.1
Valor de @vc:minVersion	Falta el atributo o está presente y tiene cualquier valor menos 1.1	1.0

Proyecto

Cuando inicie Authentic Desktop, puede abrir automáticamente el último proyecto que utilizó.

Guardar archivo

Cuando guarda un documento XML, Authentic Desktop incluye un breve comentario (`<!-- Edited with Authentic Desktop http://www.altova.com -->`) casi al principio del archivo. Esta opción solamente se puede deshabilitar si el usuario compró una licencia del producto y afecta a los archivos que se editan o guardan en las vistas Cuadrícula y Esquema.

Si el archivo XML tiene un archivo SPS asociado y marca la casilla *Authentic: guardar enlace al archivo de diseño,* el enlace al archivo SPS se guardará en el archivo XML.

Saltos de línea

Cuando abra un archivo, la codificación de caracteres para los saltos de línea del archivo se mantiene si selecciona la opción *Mantener antiguos*. Si lo prefiere, puede codificar los saltos de línea con una de estas tres codificaciones: **CR&LF** (para PC), **CR** (para MacOS) o **LF** (para Unix).

No dar formato de salida a

En la vista Texto se puede aplicar sangría a los elementos para mostrar su posición en la jerarquía de elementos (ver **Guardar archivo**). No obstante, puede invalidar esta sangría en el caso de determinados elementos. Para ello, escriba el nombre del elemento en el campo **No dar formato de salida a**. Cuando se representen estos elementos, sus descendientes no tendrán

 © 2015 Altova GmbH

espacios en blanco entre ellos (véanse los ejemplos que aparecen a continuación).

Sangría jerárquica para todos los elementos:

```
11  <xs:simpleType>
12      <xs:restriction base="xs:string">
13          <xs:maxLength value="255"/>
14      </xs:restriction>
15  </xs:simpleType>
```

El elemento `xs:restriction` se incluyó en el campo *No dar formato de salida a*:

```
11  <xs:simpleType>
12      <xs:restriction base="xs:string"><xs:maxLength value="255"/></xs:restriction>
13  </xs:simpleType>
```

Guardar y salir

Después de definir esta configuración, haga clic en **Aceptar** para terminar.

Tipos de archivo

En la pestaña ***Tipos de archivo*** puede adaptar a sus requisitos el comportamiento de Authentic Desktop dependiendo del tipo de archivo.

Elija un tipo de archivo en el cuadro de lista *Tipos de archivo* para personalizar sus funciones tal y como se describe a continuación:

Configuración del explorador de Windows

Puede definir la descripción del tipo de archivo y el tipo de contenido MIME utilizado por Windows Explorer. También puede elegir si Authentic Desktop es el editor predeterminado para los archivos de este tipo.

Conformidad

Authentic Desktop ofrece funciones especiales para determinados tipos de archivo. Para ello es necesario indicar en este panel qué tipo de gramática utiliza el tipo de archivo. En Un gran número de tipos de archivo vienen con una conformidad predeterminada adecuada para el tipo de archivo. Por ello, recomendamos que no cambie estas opciones a no ser que se trate de un tipo de archivo nuevo o que desee cambiar la conformidad del tipo de archivo intencionadamente.

Vista predeterminada

En este panel puede definir la vista predeterminada de cada tipo de archivo. La imagen anterior muestra la pestaña *Tipos de archivo* de la edición Enterprise Edition. En las demás ediciones el panel *Vista predeterminada* ofrece menos opciones de las que aparecen en la imagen.

Vista Texto

Aquí puede elegir si se usa color de sintaxis o no.

Deshabilitar validación automática

Si marca esta casilla, se deshabilita la validación automática del tipo de archivo seleccionado.

Por lo general, la validación automática se produce al abrir o guardar el archivo o cuando se cambia de vista.

Guardar elementos vacíos en formato "<E/>" abreviado

Algunas aplicaciones que usan documentos XML o documento generados por documentos XML pueden tener problemas a la hora de entender el formato abreviado de los elementos vacíos `<Element/>`, definido en la especificación XML 1.0. Por eso, en Authentic Desktop le damos la opción de guardar los elementos vacíos en el formato `<Element></Element>`, que es más largo pero también es válido.

Agregar extensión de archivo nueva

Añade un tipo de archivo nuevo a la lista *Tipos de archivo*. Después debe definir la configuración de este tipo de archivo nuevo usando las demás opciones de esta pestaña.

Eliminar extensión de archivo seleccionada

Elimina el tipo de archivo seleccionado y todas sus opciones de configuración.

Guardar y salir

Después de definir esta configuración, haga clic en **Aceptar** para terminar.

Vista

En la pestaña ***Vista*** puede personalizar la presentación de los documentos XML en Authentic Desktop.

Pretty-print

Si selecciona el comando **Edición | Pretty-Print** en la vista Texto o cambia a la vista Texto desde otra vista, se le aplicará formato pretty-print al documento XML. El formato pretty-print incluirá sangrías si marca la opción *Aplicar sangría* en este panel.

Mostrar logotipo

Si desea que la aplicación se inicie más rápido, puede desactivar la pantalla de presentación que aparece al iniciarse el programa. Además, si compró una licencia del producto, tiene la opción de no incluir el logotipo del programa, el aviso de copyright ni la información de registro en los documentos que imprima desde Authentic Desktop.

Título de la pantalla

El título de la pantalla de cada ventana de documento puede llevar el nombre del archivo solamente o el nombre completo de la ruta de acceso del archivo.

Guardar y salir

Después de definir estos configuración, haga clic en **Aceptar** para terminar.

© 2015 Altova GmbH

Codificación

En la pestaña ***Codificación*** puede configurar las opciones de codificación de archivos.

Codificación predeterminada para archivos XML nuevos

En esta lista desplegable puede elegir la codificación predeterminada para los archivos XML nuevos. Cada documento nuevo se crea con una declaración XML que contiene el valor de codificación elegido aquí. Si selecciona una codificación de dos o cuatro bytes como codificación predeterminada (es decir, UTF-16, UCS-2 o UCS-4) también puede elegir entre el orden de bytes little-endian o big-endian.

La codificación de los archivos XML ya existentes se conserva y solamente se puede cambiar con el comando **Archivo | Codificación**.

Abrir archivos XML con codificación desconocida como

Si no se puede determinar la codificación de un archivo XML o si el documento XML no especifica ninguna codificación, el archivo se abre con la codificación que elija en este cuadro combinado.

Abrir archivos no XML en

Los archivos no XML nuevos y los ya existentes se abren con la codificación que elija en este cuadro combinado. Puede cambiar la codificación del documento con el comando **Archivo | Codificación**.

Marca BOM (orden de bytes)

Los documentos que tienen una codificación de caracteres de dos o cuatro bytes se pueden guardar:

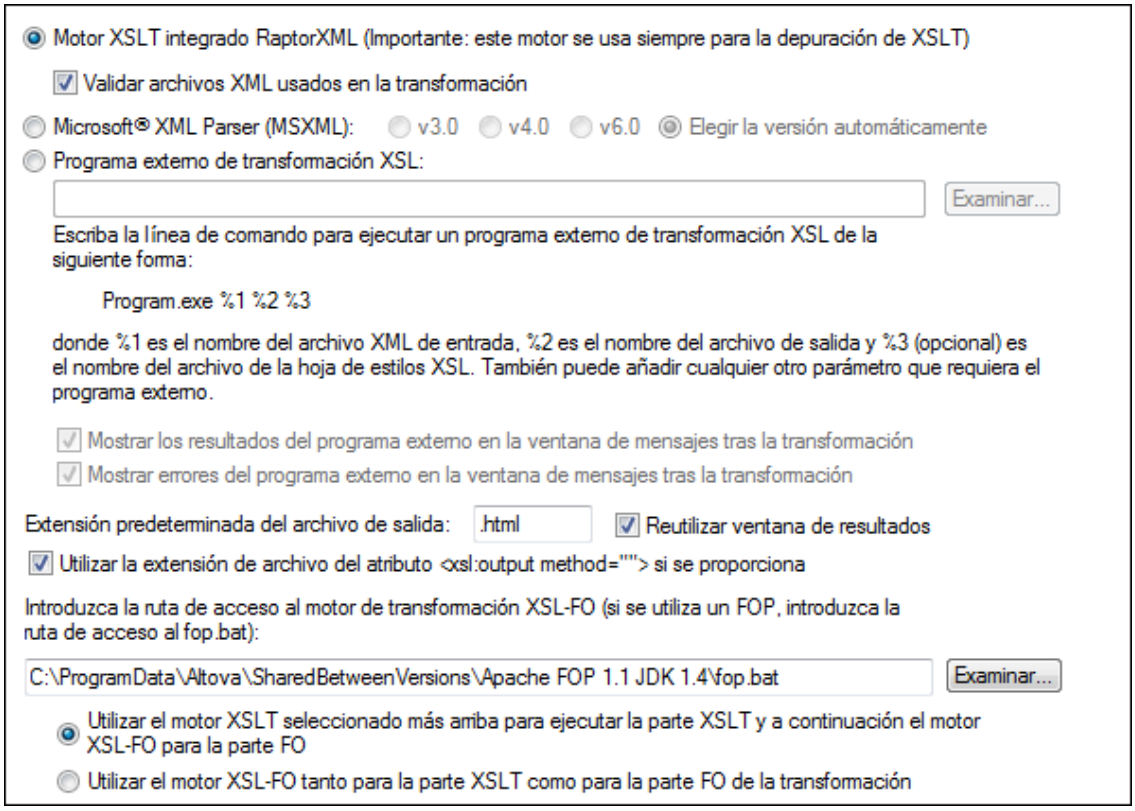
- con un orden de bytes little-endian y una marca BOM little-endian (opción *Siempre crear marca BOM (excepto para UTF-8)*) o
- con el orden de bytes detectado y la marca BOM detectada (opción *Al guardar, conservar la marca BOM detectada*).

Guardar y salir

Después de definir esta configuración, haga clic en **Aceptar** para terminar.

XSL

En la pestaña ***XSL*** (*imagen siguiente*) puede definir las opciones de configuración para las transformaciones XSL y las transformaciones XSL-FO realizadas dentro de la aplicación.


Motor XSLT integrado RaptorXML (Importante: este motor se usa siempre para la depuración de XSLT)
Validar archivos XML usados en la transformación
Microsoft® XML Parser (MSXML):
v3.0
v4.0
v6.0
Elegir la versión automáticamente
Programa externo de transformación XSL:
Examinar...
Escriba la línea de comando para ejecutar un programa externo de transformación XSL de la siguiente forma:
Program.exe %1 %2 %3
donde %1 es el nombre del archivo XML de entrada, %2 es el nombre del archivo de salida y %3 (opcional) es el nombre del archivo de la hoja de estilos XSL. También puede añadir cualquier otro parámetro que requiera el programa externo.
Mostrar los resultados del programa externo en la ventana de mensajes tras la transformación
Mostrar errores del programa externo en la ventana de mensajes tras la transformación
Extensión predeterminada del archivo de salida:
.html
Reutilizar ventana de resultados
Utilizar la extensión de archivo del atributo <xsl:output method=""> si se proporciona
Introduzca la ruta de acceso al motor de transformación XSL-FO (si se utiliza un FOP, introduzca la ruta de acceso al fop.bat):
C:\ProgramData\Altova\SharedBetweenVersions\Apache FOP 1.1 JDK 1.4\fop.bat
Examinar...
Utilizar el motor XSLT seleccionado más arriba para ejecutar la parte XSLT y a continuación el motor XSL-FO para la parte FO
Utilizar el motor XSL-FO tanto para la parte XSLT como para la parte FO de la transformación

Transformaciones XSLT

Authentic Desktop incluye los motores XSLT 1.0, XSLT 2.0 y XSLT 3.0 de Altova Raptor XML para transformaciones XSLT y para validar los archivos XML usados en las transformaciones. Dependiendo del valor del atributo `version` del elemento `xsl:stylesheet` o `xsl:transform`, en la transformación se utiliza un motor u otro.

Para transformar documentos XML con XSLT puede usar:

- El motor XSLT integrado de Altova (que incluye motores XSLT 1.0, XSLT 2.0 y XSLT 3.0).
- El analizador MSXML 3.0, 4.0 o 6.0 (que es una utilidad preinstalada). Si sabe qué versión utiliza su equipo, seleccione la opción correspondiente. Si no lo sabe, puede pedir a la aplicación que elija la versión automáticamente (*Elegir la versión automáticamente* es la opción predeterminada). En ese caso, la aplicación intenta seleccionar la versión más reciente.
- El procesador XSLT externo que usted prefiera. Para ello basta con indicar la cadena de línea de comandos que el procesador XSLT externo usa para ejecutar una transformación. Estas son las variables que puede usar para crear la cadena de línea de comandos:

  `%1` = documento XML que se debe procesar
  `%2` = archivo de salida que se debe generar
  `%3` = hoja de estilos XSLT que se debe usar (si el documento XML no contiene una referencia a la hoja de estilos)

  Por ejemplo, digamos que tiene un procesador que usa el siguiente comando para ejecutar una transformación XSLT:

 © 2015 Altova GmbH

```
myxsltengine.exe -o output.xml input.xml stylesheet.xslt parameter-
name=parameter-value
```

Entonces en Authentic Desktop puede usar las variables que se muestran arriba para generar este comando. Seleccione el botón de opción *Programa externo de transformación XSL* y escriba la siguiente línea en el cuadro de texto:

```
c:\myxsltengine\myxsltengine.exe -o %2 %1 %3 parameter-name=parameter-
value
```

Por último, si desea ver los resultados y los errores del programa externo en la ventana Mensajes de Authentic Desktop marque las casillas correspondientes.

Si marca la casilla *Reutilizar ventana de resultados*, las transformaciones que realice a partir de ahora mostrarán el documento de resultados en la misma ventana. Si el archivo XML pertenece a un proyecto y la casilla *Reutilizar ventana de resultados* está desactivada, la opción solamente surte efecto si también se desactiva la casilla *Guardar en la carpeta* de las propiedades del proyecto (*imagen siguiente*).

Transformaciones XSL-FO

Los documentos FO se procesan usando un procesador FO y la ruta de acceso del ejecutable del procesador FO debe indicarse en la casilla del motor de transformación XSL-FO. La transformación se lleva a cabo con el comando de menú **XSL/XQuery | Transformación XSL-FO**. Si el archivo de entrada (el documento que estaba activo cuando se ejecutó el comando) es un documento XSL-FO, se invoca al procesador FO para la transformación. Si el documento de entrada es un documento XML, entonces es necesario realizar primero una transformación XSLT para convertir el documento XML en un documento XSL-FO. Esta transformación XSLT se puede llevar a cabo bien con el motor XSLT que indicó como motor predeterminado (*véase más arriba*) o bien con el motor XSLT que está integrado en el procesador FO que indicó como procesador FO predeterminado para la aplicación. Seleccione cuál de estos dos motores desea usar para esta transformación.

Para terminar haga clic en **Aceptar**.

Nota: A no ser que desactive la casilla correspondiente en el programa de instalación de Authentic Desktop, el procesador FOP de Apache XML Project se instalará en la carpeta `C:\ProgramData\Altova\SharedBetweenVersions`. Y si el procesador está instalado, su ruta de acceso aparecerá automáticamente en el campo *Motor XSL-FO*. Si quiere puede cambiar esta ruta de acceso para usar otro procesador FO distinto. Sin embargo, recuerde que los demás productos de Altova que usan procesadores FO utilizarán la misma ruta de acceso (StyleVision y XMLSpy).

Guardar y salir

Después de definir esta configuración, haga clic en **Aceptar** para terminar.

Control de código fuente

En la pestaña ***Control de código fuente*** (*imagen siguiente*) puede especificar el proveedor de control de código fuente, configurarlo y definir el id. de inicio de sesión para el proveedor.

Complemento de control de código fuente

Puede elegir un complemento de control de código fuente de los sistemas de control de código fuente que tiene instalados en el equipo. Para ello seleccione una opción de la lista desplegable del cuadro combinado. Después especifique el id. de inicio de sesión. El botón **Opciones avanzadas** abre un cuadro de diálogo del sistema de control de código fuente elegido, donde puede definir opciones de configuración propias de ese control de código fuente concreto.

Preferencias de usuario

En esta pestaña también puede definir las preferencias de usuario:

- Puede realizar actualizaciones de estado en segundo plano pasado un intervalo de tiempo definido por el usuario. También puede desactivar por completo estas actualizaciones. Las bases de datos de control de código fuente de gran tamaño pueden consumir muchos recursos de red y de CPU. Pero si deshabilita las actualizaciones de estado en segundo plano o utiliza un intervalo de tiempo mayor entre actualización y actualización el sistema puede ser más rápido.
- Cuando se abre y se cierra un proyecto, los archivos se pueden desproteger y proteger (respectivamente) de forma automática.
- Puede evitar que aparezcan los cuadros de diálogo Desprotección y Protección.
- El botón **Restaurar** se habilita si marcó/activó la opción *No volver a mostrar* en alguno de los cuadros de diálogo. Al pulsar el botón **Restaurar** se rehabilita el botón/la opción *No volver a mostrar*.

Guardar y salir

Después de definir esta configuración, haga clic en **Aceptar** para terminar.

© 2015 Altova GmbH

10.9 Menú Ventanas

El menú **Ventanas** ofrece comandos estándar utilizados en la mayoría de las aplicaciones Windows con los que podrá organizar las ventanas de documento de la sesión activa de Authentic Desktop.

Las ventanas que están abiertas se pueden organizar en cascada o en mosaico. Desde este menú también puede activar o desactivar (mostrar/ocultar) las ventanas de los ayudantes de entrada y las diferentes ventanas de documento.

10.9.1 En cascada

Este comando reorganiza todos las ventanas de documento que están abiertas en forma de **cascada** (es decir, las ventanas se apilan una encima de otra).

© 2015 Altova GmbH

10.9.2 En mosaico horizontal

Este comando reorganiza todas las ventanas de documento que están abiertas en forma de **mosaico horizontal** (es decir, se pueden ver todas las ventanas a la vez y se distribuyen de forma horizontal).

10.9.3 En mosaico vertical

Este comando reorganiza todas las ventanas de documento que están abiertas en forma de **mosaico vertical** (es decir, se pueden ver todas las ventanas a la vez y se distribuyen de forma vertical).

© 2015 Altova GmbH

10.9.4 Ventana de proyecto

Este comando sirve para activar/desactivar (mostrar/ocultar) la ventana Proyecto.

Esta ventana es una ventana acoplable. Si hace clic en su barra de título y la arrastra, la ventana se desprende de su posición actual y se convierte en una ventana flotante. Haga clic con el botón secundario en la barra de título de la ventana flotante para acoplarla otra vez o para ocultarla.

10.9.5 Ventana de información

Este comando sirve para activar/desactivar (mostrar/ocultar) la ventana Información.

Esta ventana es una ventana acoplable. Si hace clic en su barra de título y la arrastra, la ventana se desprende de su posición actual y se convierte en una ventana flotante. Haga clic con el botón secundario en la barra de título de la ventana flotante para acoplarla otra vez o para ocultarla.

© 2015 Altova GmbH

10.9.6 Ayudantes de entrada

Este comando activa/desactiva (muestra/oculta) los tres ayudantes de entrada.

Los tres ayudantes de entrada son ventanas acoplables. Si hace clic en su barra de título y la arrastra, la ventana se desprende de su posición actual y se convierte en una ventana flotante. Haga clic con el botón secundario en la barra de título de la ventana flotante para acoplarla otra vez o para ocultarla.

10.9.7 Ventanas de resultados

Las ventanas de resultados son un conjunto de ventanas organizadas por pestañas, como la ventana Mensajes (que muestra mensajes de validación, etc.), la ventana Buscar en archivos o la ventana XPath (que muestra los resultados de la evaluación XPath). La posición predeterminada de este conjunto de ventanas de resultados está justo debajo de la ventana principal. Con el comando **Ventanas de resultados** puede activar/desactivar (mostrar/ocultar) las ventanas de resultados.

Las ventanas de resultados son ventanas acoplables. Si hace clic en su barra de título y la arrastra, la ventana se desprende de su posición actual y se convierte en una ventana flotante. Haga clic con el botón secundario en la barra de título de la ventana flotante para acoplarla otra vez o para ocultarla.

Para más información sobre las ventanas de resultados consulte la introducción a la interfaz gráfica del usuario de este manual.

© 2015 Altova GmbH

10.9.8 Proyecto y ayudantes de entrada

Este comando activa/desactiva (muestra/oculta) a la vez la ventana Proyecto y las ventanas de los ayudantes de entrada.

10.9.9 Activar o desactivar todas

Este comando sirve para activar o desactivar todas las ventanas acoplables:

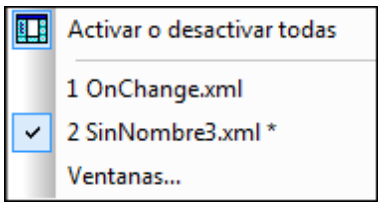
- la ventana Proyecto
- la ventana Información
- las ventanas de los tres ayudantes de entrada
- las ventanas de Resultados

Este comando es muy práctico si quiere ocultar rápidamente las ventanas que no sean de documento y conseguir el mayor espacio posible en la interfaz para ver mejor el documento en el que está trabajando.

© 2015 Altova GmbH

10.9.10 Lista de ventanas abiertas

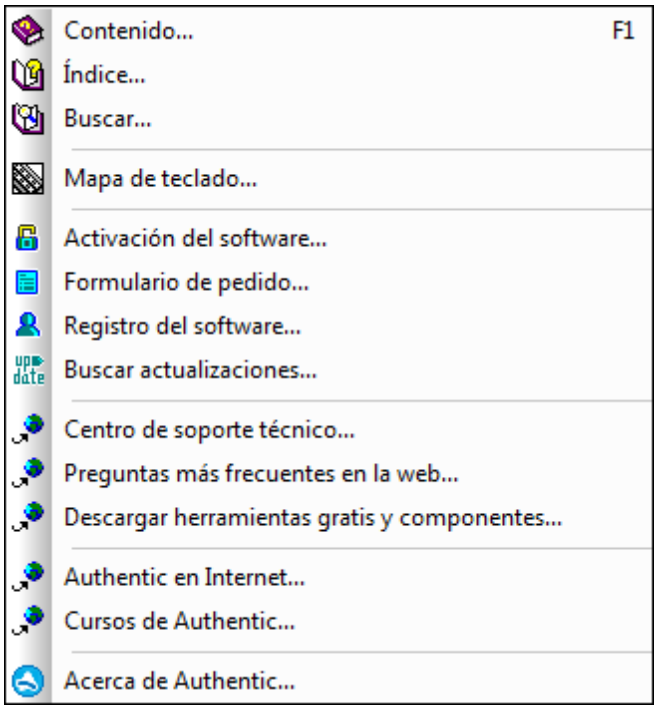
Esta lista muestra todas las ventanas que están abiertas en cada momento y permite cambiar de una ventana a otra rápidamente.



También puede usar las teclas de acceso rápido **Ctrl+Tabulador** o **Ctrl+F6** para recorrer todas las ventanas que están abiertas.

10.10 Menú Ayuda

Los comandos del menú **Ayuda** le ayudarán a obtener ayuda o información sobre Authentic Desktop y entre ellos se encuentran enlaces a páginas del sitio web de Altova.



El menú **Ayuda** también sirve para abrir el cuadro de diálogo Activación del software, donde puede introducir su código clave de licencia cuando haya comprado el producto.

© 2015 Altova GmbH

10.10.1 Contenido, Índice, Buscar...

Contenido

Descripción

Abre la ayuda en pantalla por la tabla de contenido. Esta tabla de contenido ofrece una vista general de todo el documento de ayuda. Al hacer clic en una entrada del contenido se abre el tema correspondiente de la ayuda.

Índice

Descripción

Abre la ayuda en pantalla por el índice de palabras clave. Este índice temático enumera todas las palabras claves de la ayuda y permite navegar a un tema con solo hacer doble clic en la palabra clave correspondiente. Una palabra clave puede estar asociada a varios temas.

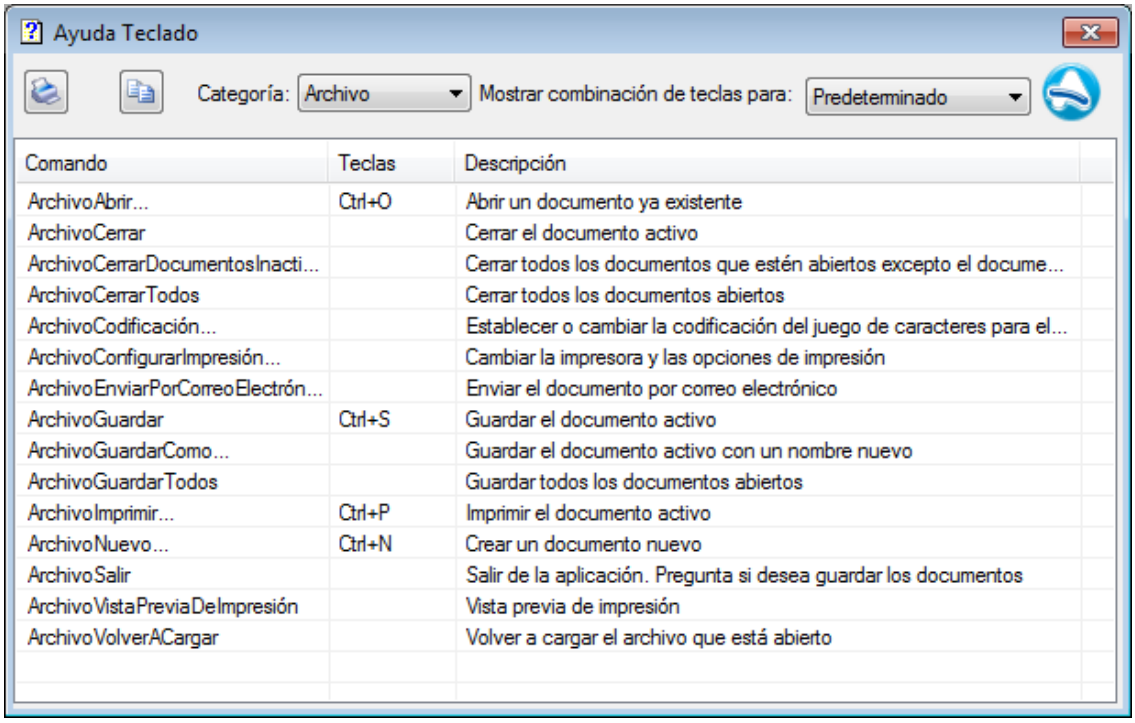
Buscar

Descripción

Abre la ayuda en pantalla por la función de búsqueda. Para buscar un término en el documento de ayuda escriba el término de búsqueda en el campo de consulta y pulse la tecla **Entrar**. Después haga doble clic en uno de los temas de la lista para ver su contenido.

10.10.2 Mapa de teclado

El comando **Ayuda | Mapa de teclado...** abre una cuadro de diálogo informativo que enumera todos los comandos de todos los menús de Authentic Desktop. Los comandos de menú van acompañados de una breve descripción y de sus teclas de acceso rápido.



Comando	Teclas	Descripción
ArchivoAbrir...	Ctrl+O	Abrir un documento ya existente
ArchivoCerrar		Cerrar el documento activo
ArchivoCerrarDocumentosInacti...		Cerrar todos los documentos que estén abiertos excepto el docume...
ArchivoCerrarTodos		Cerrar todos los documentos abiertos
ArchivoCodificación...		Establecer o cambiar la codificación del juego de caracteres para el...
ArchivoConfigurarImpresión...		Cambiar la impresora y las opciones de impresión
ArchivoEnviarPorCorreoElectrón...		Enviar el documento por correo electrónico
ArchivoGuardar	Ctrl+S	Guardar el documento activo
ArchivoGuardarComo...		Guardar el documento activo con un nombre nuevo
ArchivoGuardarTodos		Guardar todos los documentos abiertos
ArchivoImprimir...	Ctrl+P	Imprimir el documento activo
ArchivoNuevo...	Ctrl+N	Crear un documento nuevo
ArchivoSalir		Salir de la aplicación. Pregunta si desea guardar los documentos
ArchivoVistaPreviaDeImpresión		Vista previa de impresión
ArchivoVolverACargar		Volver a cargar el archivo que está abierto

Para ver los comandos de un menú determinado, seleccione el nombre del menú en el cuadro combinado *Categoría*. Si quiere imprimir el comando, haga clic en el icono en forma de impresora.

© 2015 Altova GmbH

10.10.3 Activación, formulario de pedido, registro, actualizaciones

Activación del software

Descripción

Después de descargar el software de Altova puede activarlo con un código clave de prueba gratis o con un código clave de licencia permanente.

- ***Código de prueba gratis:*** Cuando inicie el software por primera vez, aparece el cuadro de diálogo "Activación del software". En este diálogo hay un botón para solicitar un código clave de prueba gratis. Escriba su nombre, el nombre de su compañía y su dirección de correo electrónico en el diálogo y después pulse el botón **Enviar solicitud**. Nuestro servidor envía un código clave de prueba a la dirección de correo electrónico proporcionada (que suele llegar en solo unos minutos). Copie y pegue el código clave de prueba en el campo *Código clave* y pulse el botón **Aceptar** para empezar a trabajar con el software de Altova. El software permanecerá desbloqueado 30 días.
- ***Código clave permanente:*** El cuadro de diálogo "Activación del software" incluye un botón para comprar un código clave permanente. Al hacer clic en este botón se abre la página de la tienda en línea de Altova. Hay dos tipos de licencias permanentes: licencias de un solo usuario o de varios usuarios. Nuestro servidor envía la licencia a su correo electrónico. La *licencia de un solo usuario* contiene los datos de la licencia e incluye su nombre, el de su compañía, su correo electrónico y el código clave. La *licencia de varios usuarios* contiene los datos de la licencia e incluye el nombre de su compañía y el código clave. Recuerde que, según lo dispuesto en el contrato de licencia, no puede instalar más del número máximo de copias del software de Altova en equipos de su organización (licencia por número de usuarios). Asegúrese de que los datos introducidos en el cuadro de diálogo de activación del software coinciden con los datos del correo electrónico que incluye la licencia.

  Nota: cuando introduzca los datos de la licencia en el cuadro de diálogo "Activación del software", compruebe que los datos introducidos coinciden con los datos del correo electrónico que incluye la licencia. En caso de usar una licencia de varios usuarios, cada usuario debe introducir su nombre en el campo *Nombre*.

Para abrir el cuadro de diálogo "Activación del software" haga clic en el comando **Ayuda | Activación del software**.

Formulario de pedido

Descripción

Cuando desee comprar un código clave permanente, haga clic en el botón **Comprar un código clave permanente** del cuadro de diálogo "Activación del software" (*ver más arriba*) o seleccione el comando **Ayuda | Formulario de pedido**, que le llevará a la tienda en línea de Altova.

Registro del software

Descripción

Abre la página de registro del software en una pestaña del explorador web. Registro su software de Altova para recibir información sobre novedades del producto.

Buscar actualizaciones

Descripción

Busca una versión más reciente del producto en el servidor de Altova y emite un mensaje a tal efecto.

© 2015 Altova GmbH

10.10.4 Otros comandos

Centro de soporte técnico

Descripción

Es un enlace al centro de soporte técnico del sitio web de Altova. El centro de soporte técnico incluye preguntas frecuentes, foros de debate y un formulario para ponerse en contacto con el equipo de soporte técnico de Altova.

Preguntas más frecuentes

Descripción

Es un enlace a la página de preguntas frecuentes del sitio web de Altova. Esta página se actualiza constantemente con las preguntas que recibimos de nuestros clientes.

Descargar herramientas gratis y componentes

Descripción

Es un enlace al centro de descargas de componentes del sitio web de Altova. Aquí puede descargar software adicional para usarlo con los productos de Altova, como procesadores XSLT y XSL-FO y paquetes de integración. Estos componentes suelen ser totalmente gratis.

Authentic Desktop en Internet

Descripción

Es un enlace al sitio web de Altova, donde encontrará más información sobre Authentic Desktop, otros productos de Altova y tecnologías relacionadas.

Acerca de Authentic Desktop

Descripción

Abre la pantalla de presentación de la aplicación, que incluye el número de versión del producto e información sobre copyright. Si usa la versión de 64 bits de la aplicación, esto se ve en el nombre de la aplicación, que lleva el sufijo (x64). La versión de 32 bits no lleva ningún sufijo.

10.11 Línea de comandos

Algunas acciones de Authentic Desktop se pueden llevar acabo desde la línea de comandos. Se trata de estos comandos:

Abrir un archivo

Comando: `authentic.exe archivo.xml`
Acción: abre el archivo, `archivo.xml`, en Authentic Desktop

Abrir varios archivos

Comando: `authentic.exe archivo1.xml archivo2.xml`
Acción: abre los archivos, `archivo1.xml` y `archivo2.xml`, en Authentic Desktop

Asignar un archivo SPS a un archivo XML para editarlo en la vista Authentic

Comando: `authentic.exe mixml.xml /sps misps.sps`
Acción: abre el archivo XML, `mixml.xml`, en la vista Authentic con `misps.sps` como archivo SPS asociado. La marca `/sps` especifica que el archivo SPS que sigue a la marca debe usarse con el archivo XML que precede a la marca `/sps` (para poder editar el archivo XML en la vista Authentic).

Abrir una plantilla XML nueva a través de un archivo SPS

Comando: `authentic.exe misps.sps`
Acción: abre un archivo XML nuevo en la vista Authentic. La presentación del archivo se basa en el SPS y el archivo XML nuevo tendrá una estructura inicial basada en el esquema del SPS. Debe asignar un nombre al archivo XML nuevo cuando lo guarde.

© 2015 Altova GmbH

Altova Authentic 2016 Community Edition

Referencia del programador

Referencia del programador

Authentic Desktop es un servidor de automatización: expone objetos programables a otras aplicaciones, que se denominan clientes de automatización. Como resultado los clientes de automatización pueden tener acceso directo a los objetos y a las funciones que pone a su disposición el servidor de automatización. Por ejemplo, los clientes de automatización de Authentic Desktop pueden usar la función de validación XML de Authentic Desktop. Por consiguiente, los desarrolladores pueden mejorar sus aplicaciones con las funciones predefinidas de Authentic Desktop.

Los objetos programables de Authentic Desktop se ponen a disposición de los clientes de automatización por medio de la API de Authentic Desktop, que es una API de COM. En la presente documentación describimos el modelo de objetos de la API de la aplicación y todos los objetos disponibles (consulte la sección API de la aplicación).

A la API de la aplicación se puede acceder desde estos entornos:

- El editor de scripts
- Los complementos para entornos IDE
- Programas externos
- La integración con ActiveX

A continuación describimos estos entornos brevemente.

Editor de scripts: personalizar y modificar las funciones de Authentic Desktop
Puede personalizar su instalación de Authentic Desktop modificando sus funciones o agregando funciones nuevas. También puede crear formularios y modificar la interfaz del usuario añadiendo comandos de menú e iconos nuevos. Para ello basta con escribir scripts que interactúen con objetos de la API de la aplicación. A fin de ayudarle a realizar estas tareas de forma eficaz, Authentic Desktop viene con un editor de scripts integrado. Para más información sobre las funciones del editor de scripts e instrucciones de uso, consulte la sección Editor de scripts de esta referencia. Los lenguajes de programación compatibles con el editor de scripts son **JScript** y **VBScript**.

Complementos para entornos IDE: crear complementos para Authentic Desktop
En Authentic Desktop puede crear sus propios complementos e integrarlos en Authentic Desktop. Para ello puede usar la interfaz especial de Authentic Desktop para complementos. Para más información sobre cómo crear complementos, consulte la sección Complementos para entornos IDE.

Un objeto de la aplicación se pasa a la mayoría de los métodos que debe implementar el complemento para entornos IDE y la aplicación llama al objeto. Los lenguajes que se suelen utilizar para implementar un complemento para entornos IDE son **C#** y **C++**. Para más información, consulte la sección Complementos para entornos IDE.

Programas externos

También puede manipular Authentic Desktop con scripts externos. Por ejemplo, puede escribir un script para abrir Authentic Desktop a una hora concreta, abrir un archivo XML en Authentic

© 2015 Altova GmbH

Desktop, validarlo e imprimirlo. Los scripts externos utilizan la API de la aplicación para llevar a cabo estas tareas. Para más información, consulte la sección API de la aplicación.

Para usar la API de la aplicación desde fuera de XMLSpy es necesario iniciar una instancia de XMLSpy primero. La manera de iniciar la instancia de Authentic Desktop depende del lenguaje de programación utilizado. Para más información consulte el apartado Lenguajes de programación.

Básicamente Authentic Desktop se inicia por medio de su registro COM. Después se devuelve el objeto `Application` asociado con la instancia de Authentic Desktop. Dependiendo de la configuración COM, puede devolverse un objeto asociado con una instancia de Authentic Desktop que ya se esté ejecutando. Puede utilizarse cualquier lenguaje de programación que permita crear e invocar objetos COM. A continuación enumeramos los más comunes:

- Los archivos de script de JScript y VBScript tienen una sintaxis sencilla y están diseñados para acceder a objetos COM. Pueden ejecutarse directamente desde la línea de comandos de DOS o al hacer doble clic en ellos desde el explorador de Windows. Son ideales para tareas de automatización sencillas.
- C# es un lenguaje de programación con un gran número de funciones. El acceso a objetos COM se puede encapsular automáticamente con C#.
- C++ permite un control directo sobre el acceso a COM pero requiere cantidades de código relativamente grandes en comparación con otros lenguajes.
- Java: los productos de Altova vienen con clases nativas de Java, que encapsulan la API de la aplicación y ofrecen un entorno similar a Java.
- Otros lenguajes de programación que pueden ser útiles son Visual Basic for Applications, Perl y Python.

Integración con ActiveX

Otra manera de acceder a la API de la aplicación es por medio del control ActiveX de Authentic Desktop. Esta característica sólo está disponible si tiene instalado el paquete de integración de Authentic Desktop. Todos los controles ActiveX tienen una propiedad que devuelve un objeto COM equivalente para su función subyacente. El control del administrador ofrece un objeto `Application`, el control de documentos ofrece un objeto `Document` y el objeto marcador de posición, si contiene un árbol de proyecto, devuelve el objeto `Project`. Los métodos compatibles con estos objetos se describen en detalle en el apartado Interfaces de la sección API de la aplicación. Debe tener cuidado y no utilizar métodos que no tienen sentido en el contexto de la integración para el control ActiveX. Para más información consulte la sección Integración con ActiveX.

Acerca de la referencia del programador

La *Referencia del programador* de la documentación de Authentic Desktop se divide en estas secciones:

- Editor de scripts: una referencia del usuario para el entorno de scripting incluido en Authentic Desktop
- Complementos para entornos IDE: explica cómo crear complementos para Authentic Desktop
- API de la aplicación: una referencia de la API de la aplicación
- Integración para ActiveX: una guía y referencia para aprender a integrar la IGU y las funciones de Authentic Desktop usando el control ActiveX.

1 Editor de scripts

El Editor de scripts de Authentic Desktop usa los componentes del Editor de formularios de Microsoft .NET Framework y, por tanto, ofrece acceso a Microsoft .NET Framework. Esto significa que además de usar JScript y VBScript con la API de Authentic Desktop (que es un API de COM), también puede acceder a las clases de Microsoft .NET Framework y utilizarlas.

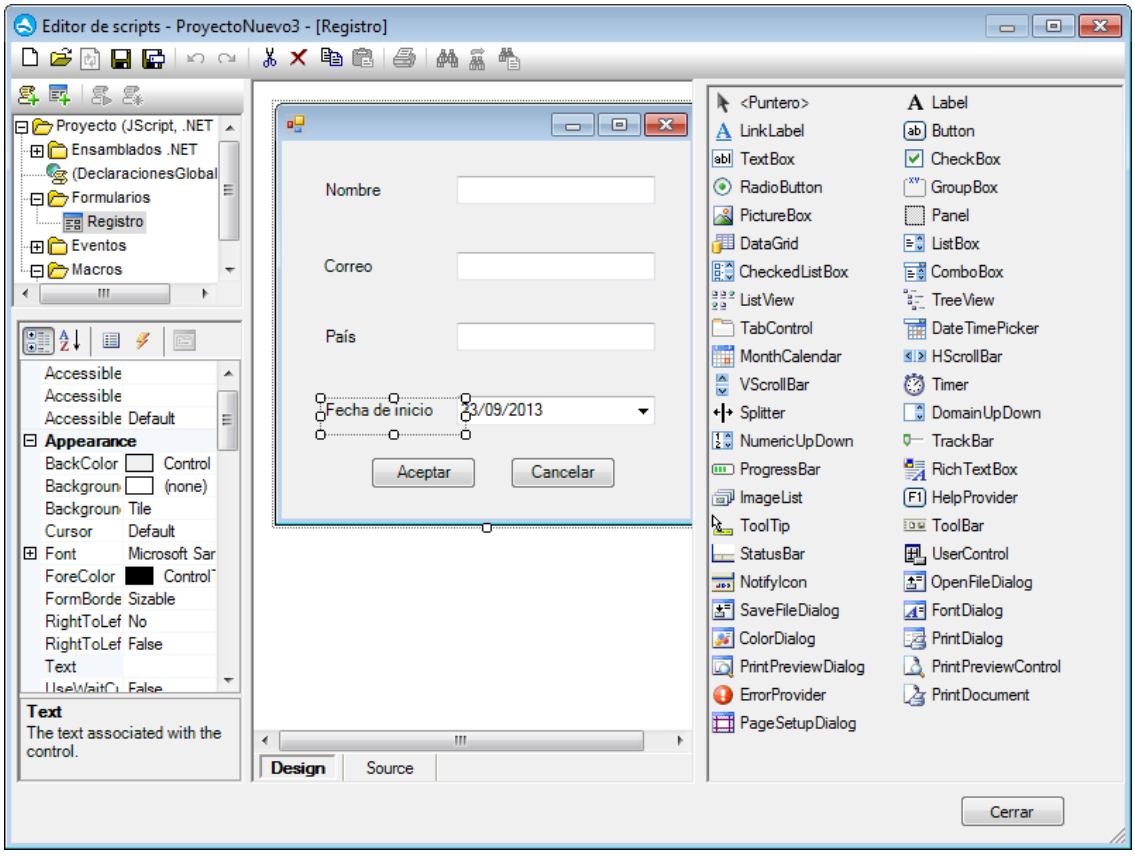
Por tanto, dentro de Authentic Desktop podrá crear y usar sus propias macros y formularios para modificar y agregar funciones a la aplicación.

Nota: el entorno de scripting **no es compatible** con el lenguaje Visual Basic. Sí lo es con VBScript y JavaScript. Por tanto, asegúrese de utilizar la sintaxis VBScript y no la sintaxis Visual Basic en el entorno de scripting.

Nota: para que funcione el Editor de scripts es necesario instalar Microsoft **.NET Framework 2.0 o superior** antes de instalar Authentic Desktop.

El Editor de scripts

El Editor de scripts (*imagen siguiente*) se abre en una ventana nueva haciendo clic en el comando de menú **Herramientas | Editor de scripts** de Authentic Desktop. Los lenguajes de programación que se pueden usar en el entorno de scripting son **JScript** y **VBScript**. El lenguaje de scripting del proyecto se puede cambiar: haga clic con el botón secundario en el nodo `Proyecto` del panel superior izquierdo del Editor y, en el menú contextual, seleccione **Configuración del proyecto**. Ahora seleccione el lenguaje pertinente en el cuadro de diálogo.



 © 2015 Altova GmbH

¿Qué puede hacer con el editor de scripts?
En el Editor de scripts puede crear formularios, controladores de eventos y macros para construir un proyecto de scripting. Los proyectos de scripting pueden ser proyectos globales para Authentic Desktop, lo cual permite usar los scripts del proyecto en la aplicación. Pero los proyectos también se pueden asignar a proyectos distintos de Authentic Desktop, lo cual permite usar scripts distintos en proyectos de Authentic Desktop diferentes.

En cada proyecto de scripting puede definir la versión del Runtime de .NET que se debe utilizar. Una aplicación puede trabajar con varios proyectos de scripting con versiones diferentes del Runtime de .NET siempre y cuando las versiones correspondientes de .NET estén instaladas. Por ejemplo: los proyectos de scripting con .NET 4.0 solo funcionan en equipos que tengan .NET 4.0 instalado.

Documentación sobre el Editor de scripts

La presente documentación del entorno de scripting se divide en varias secciones:

- Información general: resumen sobre las características del Editor de scripts y los proyectos de scripting
- Instrucciones para crear un proyecto de scripting
- Qué son las declaraciones globales
- Instrucciones para crear formularios
- Controladores de eventos de Authentic Desktop
- Instrucciones para usar macros en el Editor de scripts y en Authentic Desktop.

1.1 Información general

El Editor de scripts ofrece una interfaz gráfica donde puede (i) diseñar formularios de forma gráfica asignando scripts a los componentes del formulario, (ii) crear controladores de eventos y (iii) crear macros.

Estos formularios, controladores de eventos y macros se organizan en proyectos de scripting, que se asignan a los proyectos de Authentic Desktop y se pueden usar en la aplicación.

También puede definir variables y funciones en un script de declaraciones globales, que siempre se ejecuta antes que los scripts de macros y controladores de eventos.

Esta sección ofrece información general sobre el Editor de scripts y los proyectos de scripting y se divide en varios apartados:

- Proyectos de scripting en Authentic Desktop: describe cómo se usan en Authentic Desktop los proyectos de scripting creados con el Editor de scripts.
- Interfaz gráfica del usuario del Editor de scripts: información detallada sobre la interfaz gráfica del Editor de scripts y sus componentes.
- Componentes de un proyecto de scripting: explica qué componentes constituyen un proyecto de scripting.

Para aprender a crear los diferentes componentes del proyecto de scripting, consulte los apartados correspondientes de las secciones Declaraciones globales, Formularios, Controladores de eventos y Macros.

Ensamblados .NET

Los proyectos de scripting pueden tener referencias a ensamblados .NET (además de referencias predeterminadas). Los ensamblados .NET se pueden añadir a todo el proyecto de scripting o a las macros (usando el nuevo comando `CLR.LoadAssembly` en el código fuente, como se explica en el apartado Comandos integrados). Por ejemplo, puede añadir ensamblados desde el caché global de ensamblados.

Para añadir un ensamblado haga clic con el botón secundario en el proyecto o en una macro y, en el menú contextual, seleccione **Agregar ensamblado .NET | Caché global de ensamblados (GAC)**.

Esto funciona igual que en Visual Studio y no solo ofrece acceso a todo Microsoft .NET Framework sino también a cualquier ensamblado definido por el usuario.

© 2015 Altova GmbH

1.1.1 Proyectos de scripting en Authentic Desktop

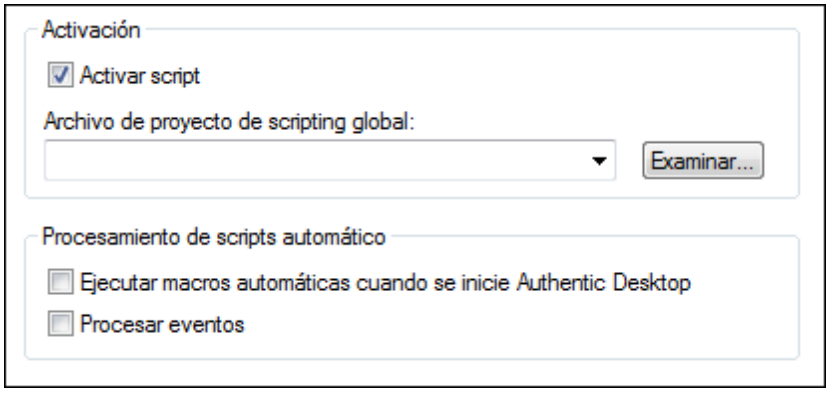
Todos los scripts creados en el Editor de scripts se almacenan en proyectos de scripting de Altova (archivos **.asprj**) y puede crear tantos proyectos de scripting de Altova como quiera. Una vez creado, el archivo .asprj puede tener varios usos:

- Se puede definir como proyecto global de scripting para Authentic Desktop. A los scripts de un proyecto global de scripting se les puede llamar desde dentro de la aplicación y las macros del proyecto global de scripting se pueden usar para todos los proyectos de Authentic Desktop.
- Se puede asignar a un proyecto de Authentic Desktop (como proyecto de aplicación). Cuando se abre un proyecto de Authentic Desktop en Authentic Desktop, se pueden invocar los scripts del proyecto de scripting asociado.

Authentic Desktop viene con un ejemplo de proyecto de scripting llamado `SampleScripts.asprj`. Este archivo está en la carpeta: `C:\Documents and Settings\<usuario>\Mis Documentos\Altova\Authentic Desktop2016\Examples\` y contiene declaraciones globales para varias tareas estándar.

Definir el proyecto de scripting global de una aplicación

El proyecto de scripting global de una aplicación se define en la pestaña *Script* del cuadro de diálogo "Opciones" de Authentic Desktop (*imagen siguiente*).

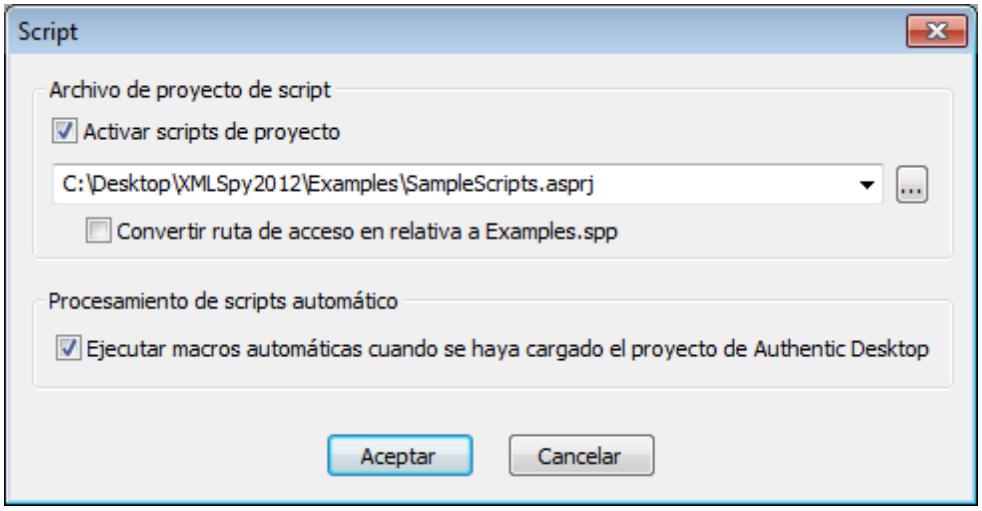


Para definir un proyecto de scripting global para Authentic Desktop, marque la casilla *Activar script* y después busque el proyecto de scripting (archivo `.asprj`) que desea activar. En esta pestaña también puede indicar: (i) si las macros automáticas del proyecto de scripting se ejecutan automáticamente cuando se inicie Authentic Desktop y (ii) si los scripts de control de eventos de la aplicación del proyecto se deben ejecutar automáticamente o no.

Nota: se pueden ejecutar scripts anidados, es decir, las macros pueden llamar a otras macros y los eventos se reciben durante la ejecución de macros o de eventos.

Para asignar un proyecto de scripting a un proyecto de Authentic Desktop:

1. En Authentic Desktop abra el proyecto de aplicación pertinente.
2. Seleccione el comando de menú **Proyecto | Configuración de script...**. Se abre el cuadro de diálogo "Script" (*imagen siguiente*).


Script
Archivo de proyecto de script
Activar scripts de proyecto
C:\Desktop\XMLSpy2012\Examples\SampleScripts.asprj
Convertir ruta de acceso en relativa a Examples.spp
Procesamiento de scripts automático
Ejecutar macros automáticas cuando se haya cargado el proyecto de Authentic Desktop
Aceptar
Cancelar

3. Marque la casilla *Activar scripts de proyecto* y seleccione el proyecto de scripting pertinente (archivo `.asprj`). Si quiere ejecutar las macros automáticas cuando se cargue el proyecto de Authentic Desktop, marque la casilla *Ejecutar macros automáticas*.
4. Para terminar haga clic en **Aceptar**.

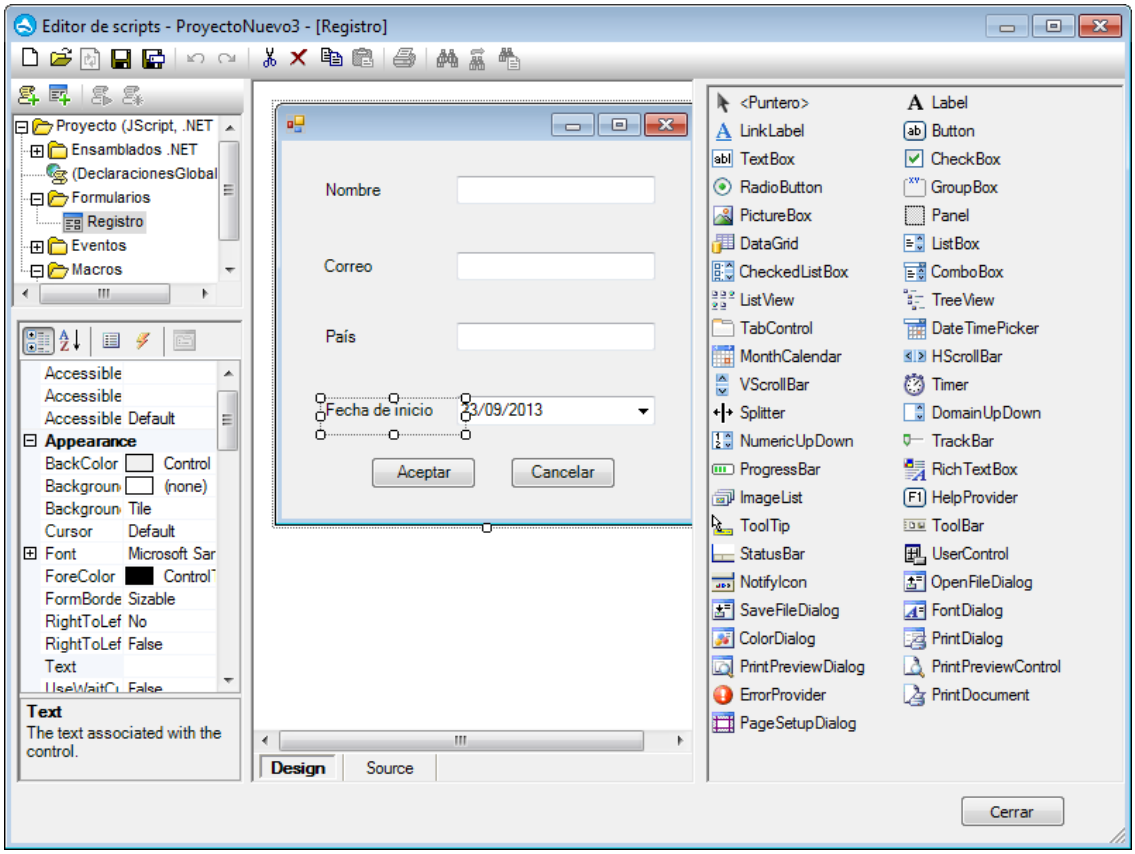
Nota: para desactivar el proyecto de scripting de un proyecto de Authentic Desktop (es decir, para anular la asignación), desactive la casilla *Activar scripts de proyecto*.

© 2015 Altova GmbH

1.1.2 La interfaz gráfica del editor de scripts

A continuación puede ver una imagen de la interfaz gráfica del usuario del Editor de scripts, que se compone de:

- Una barra de herramientas
- El panel Estructura del proyecto de scripting (arriba a la izquierda)
- El panel Propiedades y eventos (abajo a la izquierda)
- Una ventana principal con las pestañas *Design* y *Source* para el diseño y el código fuente respectivamente
- Una paleta de objetos de formulario (en el lateral derecho)



Barra de herramientas del Editor de scripts

La barra de herramientas del Editor de scripts incluye iconos para:

- Comandos de archivo como **Nuevo**, **Abrir**, **Guardar** e **Imprimir**. Estos comandos sirven para crear proyectos de scripting nuevos, abrir proyectos de scripting, guardarlos e imprimirlos.
- Comandos de edición estándar como **Copiar**, **Pegar**, **Deshacer**, **Rehacer**, **Buscar** y **Reemplazar**. Recuerde que los comandos **Buscar** y **Reemplazar** afectan al código de la pestaña *Source* del Editor de scripts.

Estructura del proyecto de scripting

La estructura del proyecto de scripting se representa en el panel superior izquierdo (*imagen siguiente*) y muestra los componentes que constituyen el proyecto de scripting. Estos

componentes pueden dividirse en cuatro categorías principales: (i) **Declaraciones globales**, (ii) **Formularios**, (iii) **Eventos** y (iv) **Macros**.

Desde el panel Estructura del proyecto puede acceder a todos los componentes del proyecto de scripting. Por ejemplo, si quiere ver y editar un formulario determinado, expanda la carpeta **Formularios** (*imagen anterior*), haga clic con el botón secundario en el formulario que le interesa y, en el menú contextual, seleccione **Abrir**.

La forma más rápida de abrir un formulario, evento, macro o script de declaraciones globales es hacer doble clic en el icono correspondiente. Para eliminar un formulario o una macro del proyecto de scripting, haga clic con el botón secundario en el componente y, en el menú contextual, seleccione **Eliminar**.

El panel Estructura del proyecto de scripting también incluye una barra de herramienta con varios iconos (*imagen siguiente*).

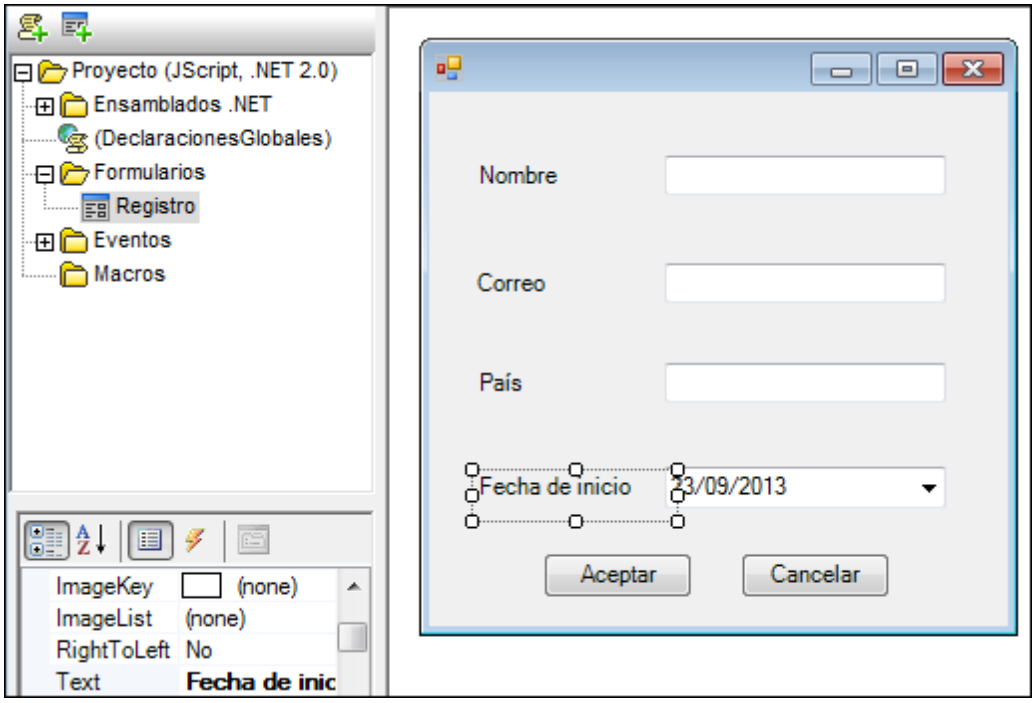
Estos iconos (de izquierda a derecha) sirven para (i) crear una macro nueva, (ii) crear un formulario nuevo, (iii) ejecutar una macro y (iv) depurar una macro. Estos comandos también están disponibles en el menú contextual que aparece al hacer clic con el botón secundario en cualquier componente de la Estructura del proyecto de scripting.

Propiedades y eventos

El panel Propiedades y eventos (*imagen siguiente*) muestra:

- Propiedades de formularios (cuando se selecciona un formulario)
- Propiedades de objetos (cuando se selecciona un objeto de un formulario, como en la imagen siguiente)
- Eventos de formularios (cuando se selecciona un formulario)
- Eventos de objetos (cuando se selecciona un objeto de un formulario)

© 2015 Altova GmbH



Para cambiar entre las propiedades y los eventos del componente seleccionado, haga clic en el icono **Propiedades** (el tercero empezando por la izquierda de la barra de herramientas del panel Propiedades y eventos) o en el icono Eventos (el cuarto icono empezando por la izquierda).

El primer y el segundo icono de la barra de herramientas empezando por la izquierda son los iconos **Por categorías** y **Alfabético** que muestran las propiedades y eventos ordenados por categorías o por orden alfabético ascendente respectivamente.

Cuando se selecciona una propiedad o evento, en la parte inferior del panel aparece una breve descripción.

Ventana principal

La ventana principal muestra el componente seleccionado y tiene una o dos pestañas, dependiendo del tipo de componente. Si se trata de un script de declaraciones globales, de un evento o de una macro, entonces aparece solo la pestaña *Source* que muestra el código fuente del componente seleccionado.

La pestaña *Source* para el código fuente ofrece estas características:

- color de sintaxis
- plegamiento de código
- creación y eliminación de marcadores con **Ctrl+F2**
- finalización automática con información sobre los parámetros
- extender e ir a la llave
- método completo o firma de la propiedad junto al ayudante de entrada de finalización automática
- resaltado de llaves durante la entrada de código

```
if ( x == y.GetName( a, b, c() ) )
```

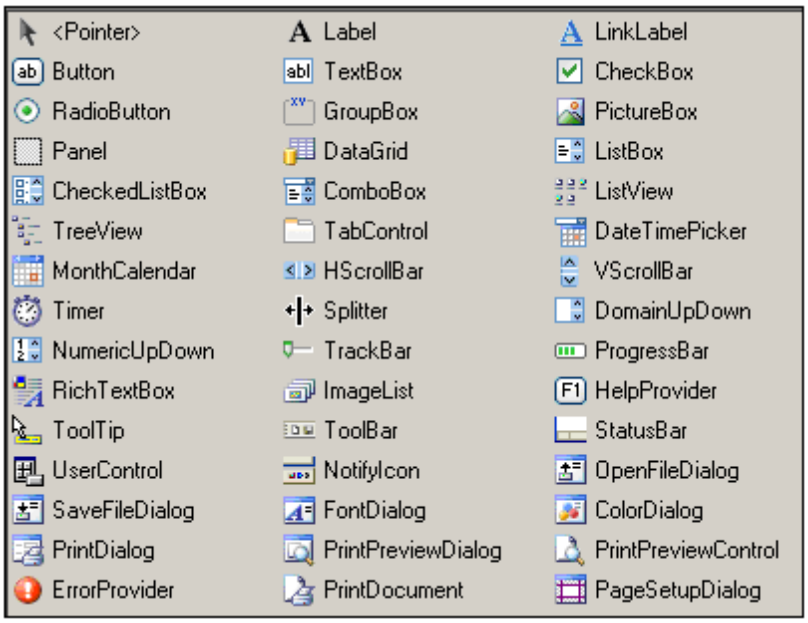
- información rápida (firma y documentación) al pasar el cursor del mouse sobre métodos o propiedades conocidas

Si la ventana principal muestra un **formulario**, entonces incluye dos pestañas: la pestaña *Design* (que muestra el formulario y permite diseñarlo) y la pestaña *Source* (que contiene el código fuente del formulario). El contenido de ambas pestañas se puede editar.

Notas:
JScript y VBScript son lenguajes sin tipo y por eso los ayudantes de entrada y la función de finalización automática solamente funciona en caso de "construcciones completas" y nombres "predefinidos".

Si los nombres empiezan por `objDocument`, `objProject`, `objXMLData` o `objAuthenticRange`, entonces aparecerán los miembros de la interfaz correspondiente. Durante la edición aparece el ayudante de entrada de finalización automática y la información sobre parámetros. No obstante, puede invocar a estos ayudantes de entrada en cualquier momento con solo pulsar **Ctrl+Barra espaciadora**.

Paleta de objetos de formulario
La paleta de objetos de formulario contiene todos los objetos que están disponibles para el diseño de formularios (*imagen siguiente*). También puede añadir controles ActiveX registrados a la paleta. Para ello haga clic con el botón secundario en la paleta y seleccione el comando **Agregar control ActiveX**.



Para insertar un objeto de la paleta en el diseño de la ventana principal, haga clic en el objeto y después en la parte del formulario donde desea insertar el objeto. El objeto se inserta en la posición elegida y en muchos casos deberá definir algunas de sus propiedades (en el panel Propiedades y eventos). Los objetos se pueden arrastrar de una parte a otra del formulario y su tamaño se puede ajustar. Además si hace clic con el botón secundario en el objeto aparece un menú contextual con varios comandos de edición (p. ej. comandos para centrar el objeto, para alinearlo, etc.).

Algunos objetos de formulario, como el temporizador `Timer`, no se añaden al formulario sino que

 © 2015 Altova GmbH

se insertan en una bandeja situada debajo de la ventana principal del Editor de scripts. Después puede seleccionar el objeto en la bandeja y definir sus propiedades y controladores de eventos en el panel Propiedades y evento.

1.1.3 Componentes de un proyecto de scripts

Los proyectos de scripting de Altova están formados por cuatro componentes principales:

- *Declaraciones globales:* este componente contiene las definiciones de variables y funciones que los scripts de formularios, macros y controladores de eventos del proyecto de scripting pueden utilizar.
- *Formularios*: este componente contiene todos los formularios definidos en el proyecto de scripting.
- *Eventos:* este componente contiene los scripts de control de eventos para todos los eventos de la aplicación (no del formulario).
- *Macros:* este componente contiene todas las macros definidas en el proyecto de scripting.

Estos componentes aparecen representados en el panel Estructura del proyecto de scripting del Editor de scripts (*imagen siguiente*) y se pueden abrir desde allí.

A continuación ofrecemos una breve descripción de cada uno de los componentes.

Declaraciones globales

El componente **DeclaracionesGlobales** es un script que contiene variables y funciones que pueden ser utilizadas por formularios, controladores de eventos y macros. Las funciones se sirven de la API de XMLSpy para acceder a las funciones de Authentic Desktop. Si crea una función o variable en el módulo **DeclaracionesGlobales** los formularios, controladores de eventos y macros del proyecto de scripting podrán acceder a la variable o función.

Para agregar una variable o función, abra el componente **DeclaracionesGlobales** (clic con el botón secundario en **DeclaracionesGlobales** en la Estructura del proyecto de scripting y después clic en **Abrir**) y edite el script de declaraciones globales en la ventana principal.

Formularios

En el Editor de scripts puede construir un formulario de forma gráfica con ayuda de la paleta de objetos de formulario, que contiene objetos como campos de entrada de texto y botones. Por ejemplo: puede crear un formulario para aceptar la entrada de un nombre de elemento para después eliminar todas las instancias de dicho elemento en el documento XML activo.

En este formulario puede asociar un script de función a un cuadro de texto que tome una variable de entrada. Y puede asociar un controlador de eventos a un botón para que empiece a ejecutar la función de eliminación, que está disponible en la API de XMLSpy. El formulario se invoca con una llamada desde dentro de una función (en el script Declaraciones globales) o desde una macro directamente. Para más información sobre cómo crear y editar formularios consulte el

 © 2015 Altova GmbH

apartado Formularios.

Control de eventos

Los scripts de control de eventos se pueden asociar a una gran variedad de eventos. Puede controlar eventos que tengan lugar dentro de formularios (eventos de formulario) y dentro de la interfaz de la aplicación (eventos de aplicación). El script asociado a un evento se ejecuta inmediatamente en cuanto el evento se desencadena.

La mayoría de los eventos tienen parámetros que ofrecen información detallada sobre el evento. El valor devuelto por el script suele indicar a la aplicación cómo debe continuar con su procesamiento (por ejemplo, puede que la aplicación no permita la edición).

Un evento se ejecuta cuando el evento relevante tiene lugar dentro del formulario o dentro de Authentic Desktop. Para más información sobre cómo crear controladores de eventos consulte el apartado Eventos.

Macros

Las macros se utilizan para implementar tareas complejas y repetitivas. Las macros no usan parámetros ni valores devueltos.

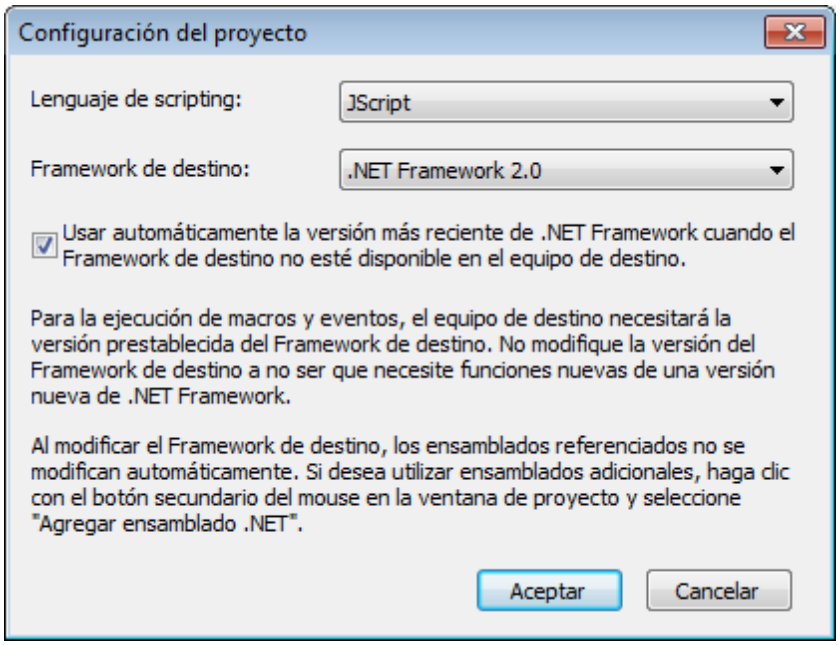
En una macro se puede acceder a todas las variables y funciones declaradas en el script de declaraciones globales y también se pueden mostrar formularios.

Para ver un ejemplo de cómo crear una macro consulte el apartado Escribir una macro. También puede consultar el apartado Ejecutar macros para aprender a llamar a una macro. Las macros se ejecutan desde la interfaz de Authentic Desktop haciendo clic en **Herramientas | Macros | *[NombreDeLaMacro]***.

1.2 Crear un proyecto de scripting

Estas son las instrucciones para crear un proyecto de scripting:

1. Abra el Editor de scripts (clic en **Herramientas | Editor de scripts**).
2. Abra un proyecto nuevo haciendo clic en el icono **Nuevo** de la barra de herramientas. Esto abre el cuadro de diálogo "Configuración del proyecto" (*imagen siguiente*). También puede abrir este cuadro de diálogo haciendo clic con el botón secundario en un proyecto del panel Estructura del proyecto de scripting (panel superior izquierdo) y seleccionando **Configuración del proyecto**.


Configuración del proyecto
Lenguaje de scripting:
JScript
Framework de destino:
.NET Framework 2.0
Usar automáticamente la versión más reciente de .NET Framework cuando el Framework de destino no esté disponible en el equipo de destino.
Para la ejecución de macros y eventos, el equipo de destino necesitará la versión prestablecida del Framework de destino. No modifique la versión del Framework de destino a no ser que necesite funciones nuevas de una versión nueva de .NET Framework.
Al modificar el Framework de destino, los ensamblados referenciados no se modifican automáticamente. Si desea utilizar ensamblados adicionales, haga clic con el botón secundario del mouse en la ventana de proyecto y seleccione "Agregar ensamblado .NET".
Aceptar
Cancelar

   En el primer cuadro combinado seleccione **JScript** o **VBScript** y en el segundo seleccione la versión de .NET Framework. Para habilitar versiones más recientes de .NET Framework (como .NET Framework 4.5 de Windows 8), marque la casilla *Usar automáticamente la versión más reciente de .NET Framework*. Haga clic en **Aceptar** para crear el proyecto de scripting nuevo.
3. Haga clic en el icono **Guardar** de la barra de herramientas para guardar el proyecto de scripting como archivo `.asprj`.
4. El proyecto de scripting está formado por varios componentes. Estos componentes suelen ser: declaraciones globales, formularios, eventos y macros. Se pueden crear en cualquier orden, pero debe entender cómo colaboran todos estos componentes. Más abajo encontrará información sobre cómo se invoca y ejecuta cada tipo de componente. Para aprender a crear un componente de cada tipo, consulte las secciones dedicadas a cada tipo de componente.
5. Cuando termine de crear todos los componentes del proyecto, haga clic en el icono **Guardar** de la barra de herramientas para guardar el proyecto de scripting en memoria.
6. Cierre el Editor de scripts.

Nota: cuando quiera cambiar el lenguaje de scripting haga clic con el botón secundario en la carpeta **`Proyecto`** y seleccione **Configuración del proyecto**.

Cómo se invocan y ejecutan los formularios, los controladores de eventos y las macros

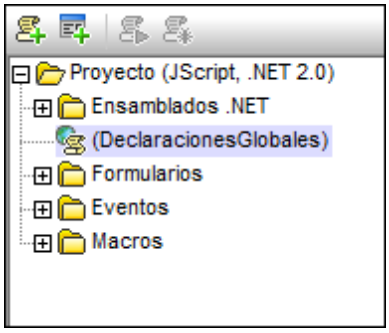
© 2015 Altova GmbH

Los formularios, controladores de eventos y macros se crean en el Editor de scripts. No obstante, no todos ellos se invocan y ejecutan de la misma manera. Esto debe tenerse en cuenta a la hora de crear el proyecto de scripting.

- Los formularios se invocan con una llamada desde dentro de una función del script de declaraciones globales o de una macro directamente.
- Los controladores de eventos se ejecutan cuando el evento correspondiente ocurre en Authentic Desktop. Si el controlador de un evento está definido tanto en el proyecto de scripting global como en el proyecto de scripting de un proyecto de Authentic Desktop, primero se ejecuta el controlador de eventos del proyecto de scripting del proyecto y después el del proyecto de scripting global.
- Las macros se ejecutan desde la interfaz de Authentic Desktop haciendo clic en **Herramientas | Macros | *[NombreDeLaMacro]***. En una macro se puede tener acceso a todas las funciones y variables declaradas en el script de declaraciones globales y mostrar los formularios de entrada del usuario.

1.3 Declaraciones globales

El componente **DeclaracionesGlobales** está presente por defecto en todos los proyectos de scripting (*imagen siguiente*) y, por tanto, no hace falta crearlo. Para poder agregar variables y funciones al script de declaraciones globales, debe abrir el script de declaraciones globales y añadirle el fragmento de código correspondiente. Para más información consulte el apartado Componentes de un proyecto de scripting y Crear un proyecto de scripting.



Para abrir un script de declaraciones globales de un proyecto de scripting, primero haga clic con el botón secundario en el nodo **DeclaracionesGlobales** de la Estructura del proyecto de scripting (*imagen anterior*). Después seleccione **Abrir** y el script de declaraciones globales se abre en la ventana principal del Editor de scripts.

Nota: cada vez que se ejecuta una macro o que se llama a un controlador de eventos, las declaraciones globales se vuelven a inicializar.

A continuación puede ver un ejemplo de función. Recuerde que si crea una variable o función en el script de declaraciones globales, los formularios, controladores de eventos y macros podrán acceder a esa variable o función.

Ejemplo de función

La función de nuestro ejemplo se llama `QuitarTodosEspaciosDeNombres` y este es su código:

```
function QuitarTodosEspaciosDeNombres(objXMLData)
{
       if(objXMLData == null)
    return;

  if(objXMLData.HasChildren)       {
    var objChild;

    // spyXMLDataElement := 4
    objChild = objXMLData.GetFirstChild(4);

    while(objChild)  {
    QuitarTodosEspaciosDeNombres(objChild);

          try {
              var nPos,txtName;
              txtName = objChild.Name;
```

```
                if((nPos = txtName.indexOf(":")) >= 0)    {
                    objChild.Name = txtName.substring(nPos+1);
                }

        objChild = objXMLData.GetNextChild();
              }
              catch(Err)              {
                  objChild = null;
              }
           }
        }
     }
```

Nota:

- Puede definir variables locales y funciones auxiliares dentro de macros y controladores de eventos. Por ejemplo:

```
//valor devuelto: true permite la edición
//valor devuelto: false no permite la edición
var txtLocal;
function Helper()
{
  txtMessage = txtLocal;
  Application.ShowForm("MsgBox");
}
function On_BeforeStartEditing(objXMLData)
{
  txtLocal = "On_BeforeStartEditing()";
  Helper();
}
```

- El uso de funciones recursivas es compatible.

1.4 Formularios

El proceso de creación y edición de formularios en el Editor de scripts comprende estos pasos:

1. Crear un formulario nuevo: el formulario nuevo se crea, se le pone un nombre y se definen sus propiedades.
2. Diseñar el formulario: el formulario se diseña añadiéndole objetos de formulario y asignando valores a estos objetos.
3. Crear scripts de eventos de formulario: se asignan scripts a los eventos del formulario.

© 2015 Altova GmbH

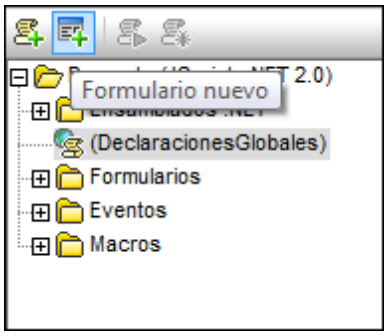
1.4.1 Crear un formulario nuevo

El proceso de creación de formularios en el Editor de scripts comprende estos pasos:

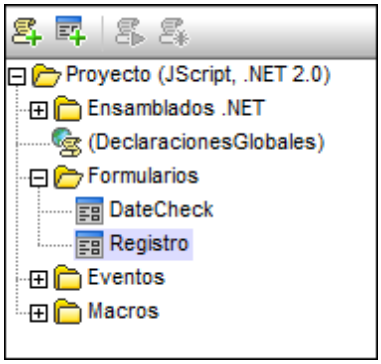
1. Crear un formulario nuevo y darle un nombre
2. Definir las propiedades del formulario

Crear un formulario nuevo y darle un nombre

Para agregar un formulario nuevo a un proyecto de scripting, haga clic en el icono **Agregar formulario** (*imagen siguiente*) de la barra de herramientas del panel Estructura del proyecto. El formulario nuevo se crea y puede escribir el nombre del formulario.

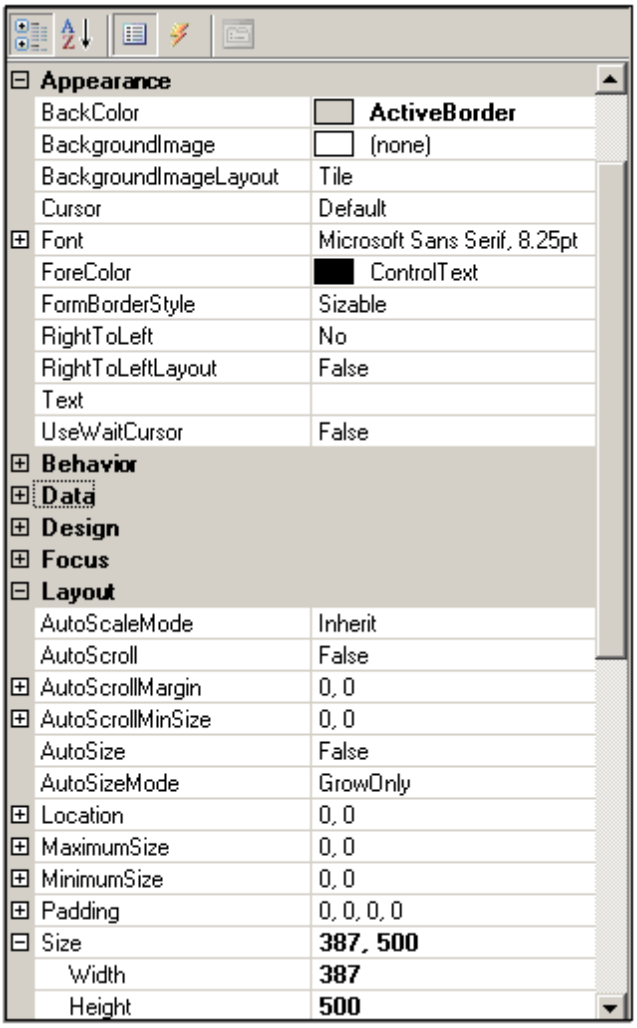


El formulario nuevo se añade al proyecto y aparece en la ventana principal. En el panel Estructura del proyecto aparece el nuevo formulario bajo la carpeta **`Formularios`**. Pulse la tecla **F2** para cambiar el nombre al formulario. También puede hacer clic con el botón secundario en el formulario y después en **Cambiar de nombre** en el menú contextual. En el ejemplo de la imagen siguiente creamos un formulario nuevo llamado **`Registro`**.



Propiedades de los formularios

Las propiedades del formulario (p. ej. su tamaño, color de fondo, fuente, etc.) se pueden definir en el panel Propiedades. En la imagen siguiente, por ejemplo, puede ver las propiedades de las categorías *Layout* (disposición) y *Appearance* (aspecto).



Probar un formulario

Para probar un formulario haga clic con el botón secundario en el formulario en el panel Estructura del proyecto. En el menú contextual seleccione el comando **Probar formulario**.

© 2015 Altova GmbH

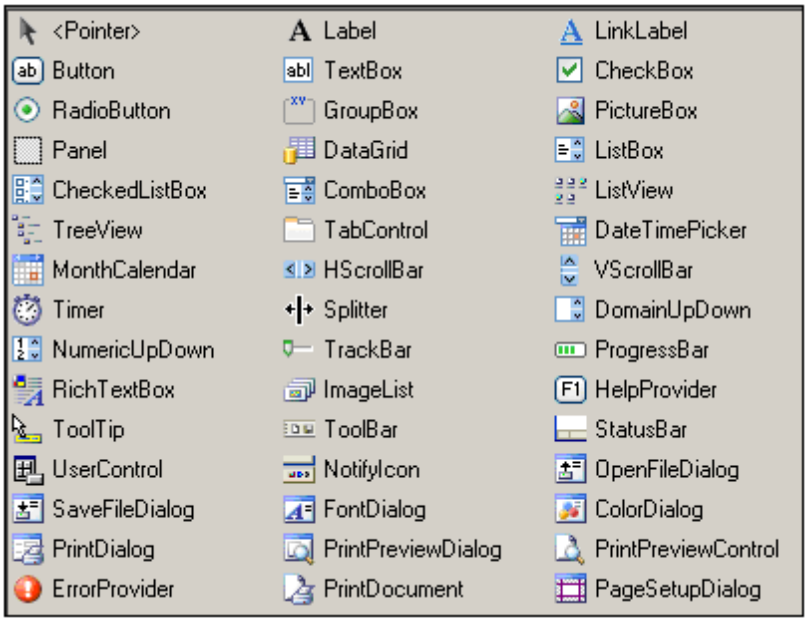
1.4.2 Diseño de formularios y objetos

El proceso de diseño de formularios en el Editor de scripts comprende estos pasos:

- Primero se colocan objetos de la paleta de objetos de formulario en el diseño del formulario.
- Después se asignan valores a las propiedades de los diferentes objetos del formulario.
- Por último se asignan scripts a los eventos del formulario.

La paleta de objetos de formulario

La paleta de objetos de formulario contiene todos los objetos que están disponibles para el diseño de formularios (*imagen siguiente*). También puede añadir controles ActiveX registrados a la paleta. Para ello haga clic con el botón secundario en la paleta y seleccione el comando **Agregar control ActiveX**.



Para insertar un objeto de la paleta en el diseño de la ventana principal, haga clic en el objeto y después en la parte del formulario donde desea insertar el objeto. El objeto se inserta en la posición elegida y en muchos casos deberá definir algunas de sus propiedades (en el panel Propiedades y eventos). Los objetos se pueden arrastrar de una parte a otra del formulario y su tamaño se puede ajustar. Además si hace clic con el botón secundario en el objeto aparece un menú contextual con varios comandos de edición (p. ej. comandos para centrar el objeto, para alinearlo, etc.).

Algunos objetos de formulario, como el temporizador `Timer`, no se añaden al formulario sino que se insertan en una bandeja situada debajo de la ventana principal del Editor de scripts. Después puede seleccionar el objeto en la bandeja y definir sus propiedades y controladores de eventos en el panel Propiedades y evento.

Estos son los objetos más utilizados:

***Label* (etiqueta):** añade campos de texto que se usan como título o descripción

***Button* (botón):** añade un botón. A estos botones puede asignar mapas de bit como imágenes de fondo.

***Check Box* (casilla):** añade una casilla de verificación que permite realizar selecciones de tipo `Sí/No`.

***Combo Box* (cuadro combinado):** añade un cuadro combinado, que permite al usuario seleccionar una opción de un menú desplegable.

***List Box* (cuadro de lista):** añade un cuadro de lista, que enumera todos los elementos que se pueden seleccionar.

***TextBox* (cuadro de texto):** permite al usuario escribir una sola línea de texto.

***Rich TextBox* (cuadro de texto enriquecido):** permite al usuario escribir varias líneas de texto.

Crear objetos y configurar sus propiedades

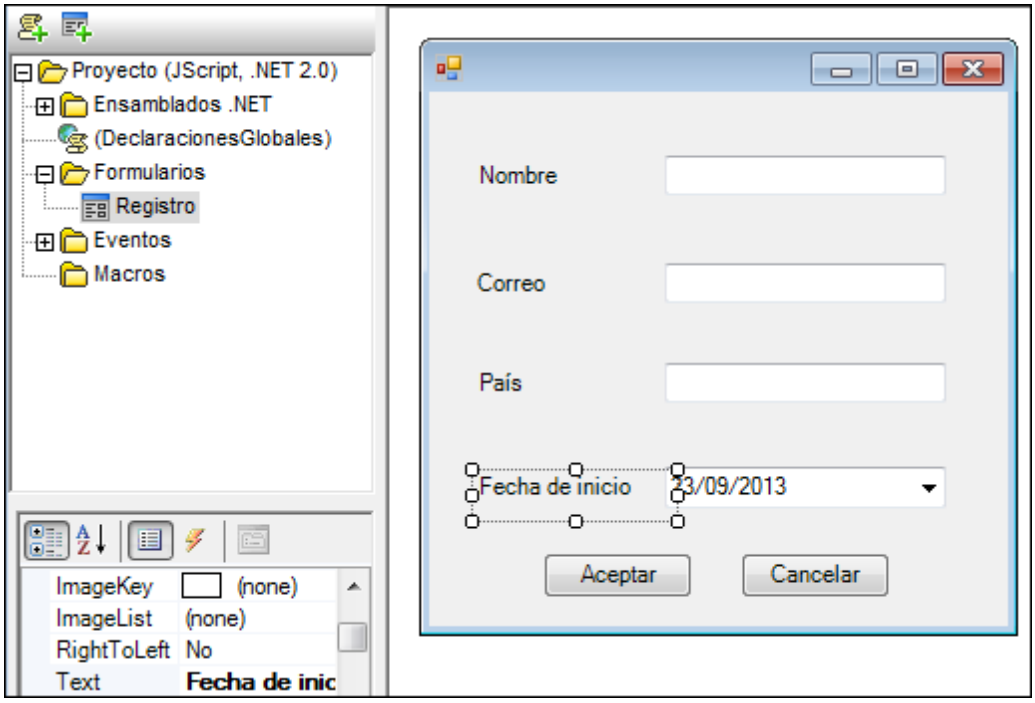
Para crear un objeto en el formulario primero debe seleccionar el objeto en la paleta de objetos y después debe hacer clic en la parte del formulario donde desea insertar el objeto. Una vez insertado, el objeto puede arrastrarse a otra posición del formulario o puede ajustar su tamaño.

Cuando se selecciona un objeto del formulario sus propiedades aparecen en el panel Propiedades y eventos, donde las puede configurar. Puede hacer clic en el icono **Propiedades** de la barra de herramientas de este panel para ver una lista de propiedades del objeto.

Por ejemplo, en la imagen siguiente seleccionamos el objeto **Etiqueta** que tiene el texto *Fecha de inicio*. En el panel Propiedades y eventos el nombre del objeto (el que se usa para identificar el objeto en el código: `Etiqueta1`) se define en la categoría de propiedades *Design*. En este caso el objeto se llama `Etiqueta1`.

El texto de la etiqueta (que es lo que aparece en el formulario) debe introducirse como valor de la propiedad *Texto* de la categoría de propiedades *Appearance*.

© 2015 Altova GmbH



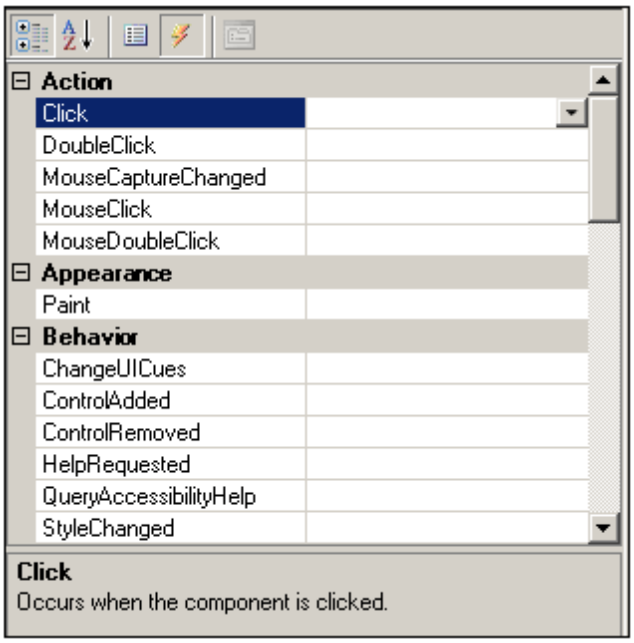
Para asignar otras propiedades al objeto, introduzca el valor correspondiente en las propiedades del panel Propiedades y eventos.

Probar un formulario

Para probar un formulario haga clic con el botón secundario en el formulario en el panel Estructura del proyecto. En el menú contextual seleccione el comando **Probar formulario**.

1.4.3 Eventos de formularios

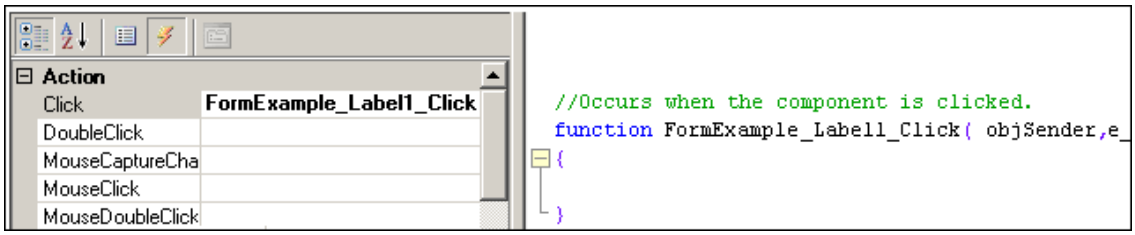
Cuando está seleccionado un objeto en el diseño, haga clic en el icono **Eventos** de la barra de herramientas del panel Propiedades y eventos (cuarto icono empezando por la izquierda) para ver todos los eventos que están disponibles para el objeto (*imagen siguiente*). Los eventos pueden aparecer ordenados por categoría (*imagen siguiente*) o alfabéticamente.



Por cada evento puede introducir el nombre de un controlador de eventos o función ya existente. También tiene estas dos opciones:

- Puede hacer doble clic en un evento para crear (i) un script de función vacío en la pestaña *Source* de la ventana principal y (ii) una asociación entre la función recién creada y el evento seleccionado.
- Puede hacer doble clic en un botón de la pestaña *Design* para generar código auxiliar vacío para el evento en la pestaña *Source*.

La captura de pantalla siguiente, por ejemplo, se tomó después de hacer doble clic en el evento **`Click`**. Observe que en la ventana principal se creó una función de controlador de eventos vacía llamada `FormExample_Label1_Click` y que en el panel Propiedades y eventos esta función se asoció al evento **`Click`**.



Ahora podemos insertar el código necesario y guardar el proyecto.

Escribir scripts

Cuando termine de diseñar el formulario de forma visual, los objetos del formulario estarán

© 2015 Altova GmbH

asociados a los correspondientes scripts. El ejemplo de script que aparece a continuación añade colores cuando se hace clic en un botón. El script se inserta como controlador de eventos para el evento `Click` del botón `Botón1` (el evento está disponible en el panel Propiedades y eventos cuando se selecciona el botón en el diseño):

```
function FormExample_Botón1_Click( objSender, e_EventArgs )
{
	// Establece el color de primer plano (rojo) del botón
	objSender.ForeColor = CLR.Static( "System.Drawing.Color" ).Red;
	// Establece el color de fondo (azul) del botón
	objSender.BackColor = CLR.Static( "System.Drawing.Color" ).Blue;
	// Establece el color de fondo (verde) del formulario
	objSender.FindForm().BackColor =
CLR.Static( "System.Drawing.Color" ).Green;
```

1.5 Eventos

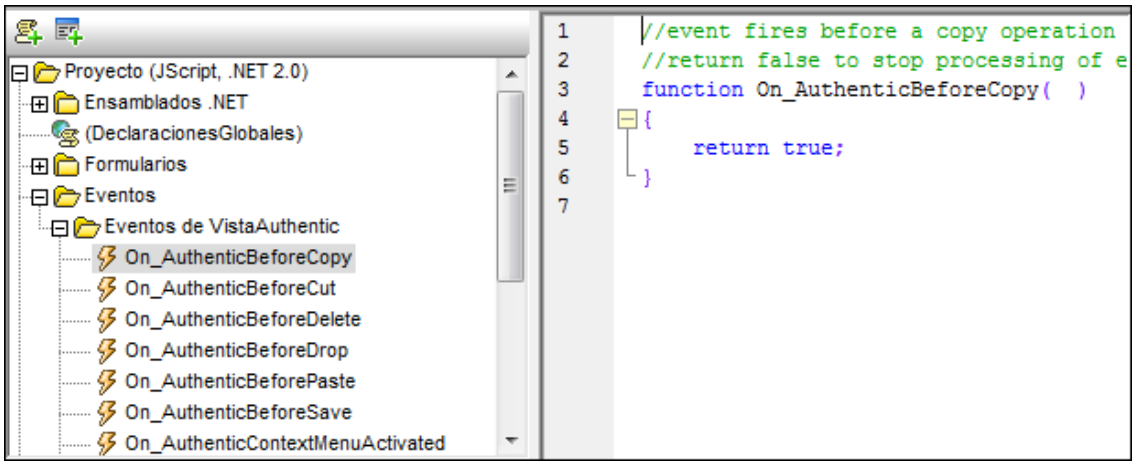
El componente **Eventos** del proyecto de scripting (*imagen siguiente*) contiene carpetas para todos estos eventos:

- Eventos de aplicación
- Eventos de documento
- Eventos de la vista Authentic
- Eventos de la vista Cuadrícula
- Eventos de la vista Texto

No olvide que todos estos eventos son propios de Authentic Desktop, no como los eventos de formularios. Cada una de estas carpetas de eventos contiene un conjunto de eventos para los que puede escribir scripts de controladores de eventos.

Por ejemplo, en la imagen siguiente puede ver los eventos de aplicación.

Para acceder al script de control de cualquiera de estos eventos, haga clic con el botón secundario en el evento y seleccione el comando **Abrir** del menú contextual. El script aparece en la ventana principal (*imagen siguiente*), donde puede editarlo. Cuando termine de editar el script guarde los cambios haciendo clic en el icono **Guardar** de la barra de herramientas del Editor de scripts.



Además debe tener en cuenta que:

- El nombre del evento debe estar correctamente escrito en el encabezado de función. De lo contrario no se podrá llamar al controlador de eventos.
- Dentro de las macros y de los controladores de eventos se pueden definir variables locales y funciones auxiliares. Por ejemplo:

```
//valor devuelto: true permite la edición
//valor devuelto: false no permite la edición
var txtLocal;
function Helper()
{
  txtMessage = txtLocal;
  Application.ShowForm("MsgBox");
}
function On_BeforeStartEditing(objXMLData)
{
```


© 2015 Altova GmbH

```
  txtLocal = "On_BeforeStartEditing()";
  Helper();
}
```

- Para que los eventos se procesen debe activar la opción *Procesar eventos* del cuadro de diálogo "Opciones".
- Puede obtener más información en el apartado Notas sobre programación.

Eventos de aplicación

OnInitialize
El evento `OnInitialize` se desencadena después de que aparezca la ventana principal pero antes de que se cargue un proyecto. Este evento no se desencadena si la aplicación no se puede cargar.

OnRunning
Si la aplicación está completamente cargada, el evento `OnRunning` se desencadena después de que tenga lugar el evento `OnInitialize`.

OnShutdown
Este evento se desencadena después de que se cierren los proyectos y documentos que estaban abiertos cuando se cerró la aplicación. La ventana principal deja de estar visible.

Ejemplo

El script que aparece a continuación es un controlador de eventos para el evento `On_BeforeOpenProject`. Este script permite añadir un script que se ejecutará antes de que Authentic Desktop abra un proyecto. El script abre uno por uno todos los archivos XML situados en la carpeta XML del proyecto y los valida. Si un archivo no es válido, el script muestra el error de validación y se detiene. Si un archivo es válido, se cierra y el script abre el archivo siguiente.

Escriba este script para el evento `On_BeforeOpenProject()` y después guarde el proyecto de scripting.

```
function On_BeforeOpenProject()
{
 var bOK;
 var nIndex,nCount;
 var objItems,objXMLFolder = null;

 objItems = Application.CurrentProject.RootItems;
 nCount = objItems.Count;

 // Buscar la carpeta XML
 for(nIndex = 1;nIndex <= nCount;nIndex++)  {
  var txtExtensions;
  txtExtensions = objItems.Item(nIndex).FileExtensions;

 if(txtExtensions.indexOf("xml") >= 0)      {
  objXMLFolder = objItems.Item(nIndex);
 break;
 }
}
```

```
// ¿Existe la carpeta XML?
if(objXMLFolder){
 var objChild,objDoc;

 nCount = objXMLFolder.ChildItems.Count;

 // Recorrer los archivos xml asociados
 for(nIndex = 1;nIndex <= nCount;nIndex++)  {
  objChild = objXMLFolder.ChildItems.Item(nIndex);

  try{
   objDoc = objChild.Open();

   // Usar el método JScript para acceder a parámetros out
   var strError = new Array(1);
   var nErrorPos = new Array(1);
   var objBadData = new Array(1);

   bOK = objDoc.IsValid(strError,nErrorPos,objBadData);

   if(!bOK) {
    // Si no supera la validación, debemos mostrar el
    // mensaje de XMLSpy
    // pero tenemos que crear el formulario "CuadroMje" y
    // definir la variable global MensajeTxt
    //
    // MensajeTxt = Position:" + nErrorPos[0] + "\n" + // strError[0];
    // MensajeTxt +="\n\nXML:\n" + objBadData[0].Name + ", " +
    //    objBadData[0].TextValue;
    //
    // Application.ShowForm("CuadroMje");

    break;
   }

   objDoc.Close(true);
   objDoc = null;
  }
  catch(Err) {
   // Aquí es buena idea mostrar la descripción del error

   // MensajeTxt = Err.Description;
   // Application.ShowForm("CuadroMje");

   break;
   }
  }
 }
}
```


© 2015 Altova GmbH

Probar el controlador de eventos

Cierre el Editor de scripts y abra cualquier proyecto en Authentic Desktop para ver cómo se controla el evento `BeforeOpenProject`.

1.6 Macros

Las macros sirven para automatizar tareas repetitivas o complejas. En el entorno de scripting puede crear un script que llame a funciones de la aplicación y a otras funciones personalizadas definidas por el usuario. Es decir, las macros ofrecen un potente método de automatizar tareas dentro de Authentic Desktop. Esta sección se divide en varios apartados:

- Crear y editar una macro: instrucciones para crear una macro nueva y editar macros ya existentes.
- Ejecutar una macro: instrucciones para ejecutar una macro desde el Editor de scripts y desde el entorno de Authentic Desktop.
- Depurar una macro: instrucciones para depurar una macro.

Nota importante

Es muy importante tener en cuenta estos puntos:

- En el proyecto de scripting activo puede incluir tantas macros como quiera. Todas ellas se guardan en el archivo de proyecto de scripting de Altova (archivo `.asprj`).
- Las funciones que se usan en una macro se pueden guardar como declaración global. Todas las declaraciones globales se guardan en el archivo de proyecto de scripting de Altova (archivo `.asprj`).
- Puede probar las macros ejecutándolas dentro del Editor de scripts. En el editor también puede depurar las macros.
- Authentic Desktop puede tener un proyecto de scripting global, un proyecto de scripting adicional asignado al proyecto activo y todas las macros que estén disponibles en ambos proyectos. Consulte el apartado Ejecutar una macro para obtener más información.

© 2015 Altova GmbH

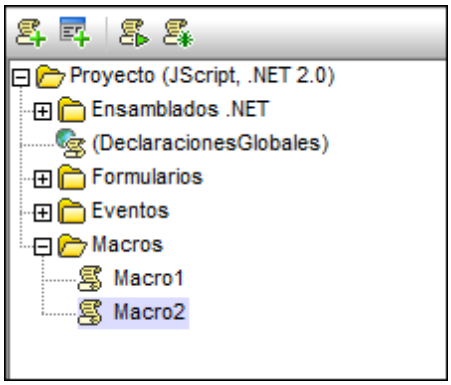
1.6.1 Crear y editar una macro

A continuación describimos las operaciones necesarias para crear una macro nueva y editar macros ya disponibles.

Crear una macro nueva

Haga clic con el botón secundario en la carpeta Macros del panel Estructura del proyecto de scripting y seleccione **Agregar macro** en el menú contextual. En realidad este comando siempre está disponible en el menú contextual que aparece al hacer clic con el botón secundario en cualquier elemento del panel. Otra opción es hacer clic en el icono **Macro nueva** de la barra de herramientas del panel Estructura del proyecto de scripting.

El documento de macro recién creado (y vacío) aparece en la ventana principal del editor y el nombre de la macro aparece en la barra de título (*imagen siguiente*).



Poner un nombre a la macro o cambiarle el nombre

Para dar un nombre a una macro o cambiar su nombre, haga clic en el nombre de la macro en el panel Estructura del proyecto de scripting y pulse la tecla **F2**. También puede hacer clic con el botón secundario en la macro y seleccionar **Cambiar el nombre de la macro** en el menú contextual.

Abrir una macro

Para abrir una macro haga clic con el botón secundario en la macro en el panel Estructura del proyecto (*imagen anterior*) y seleccione **Abrir** en el menú contextual. La macro se abre en la ventana principal y su nombre aparece en la barra de título del Editor de scripts (*imagen siguiente*). También puede abrir la macro haciendo doble clic en el nombre de la macro en el panel Estructura del proyecto.

Editar la macro

Para editar una macro escriba o edite su código en la ventana principal del editor. Por ejemplo, el código que aparece a continuación crea en memoria el formulario `Formulario1` y lo muestra. El `Formulario1` debe haberse creado de antemano (con el Editor de scripts).

```
objForm = CreateForm( 'Formulario1' );
objForm.ShowDialog();
```

Esta macro utiliza la función `QuitarTodosEspaciosDeNombres` para quitar todos los espacios de nombres del documento XML activo.

```
if(Application.ActiveDocument != null)    {

QuitarTodosEspaciosDeNombres(Application.ActiveDocument.RootElement);
              Application.ActiveDocument.UpdateViews();
}
```

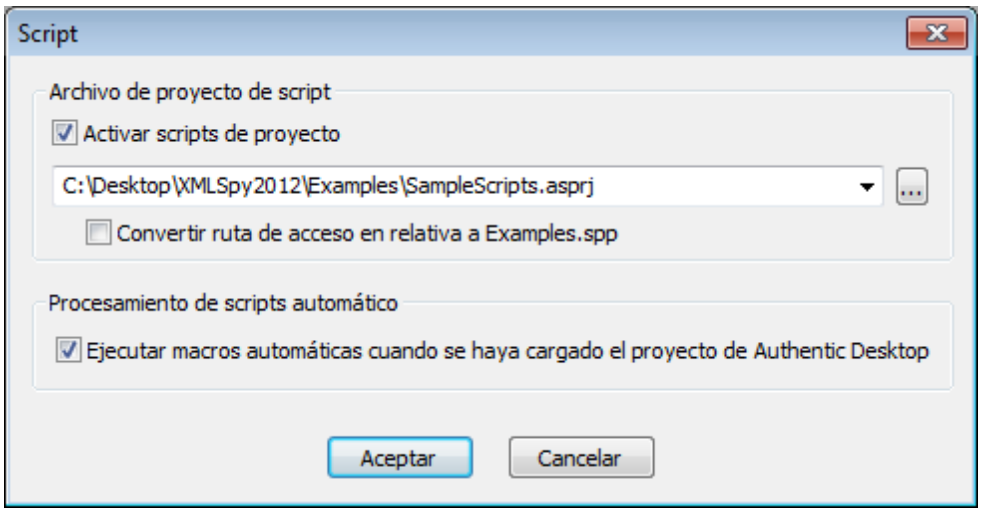
La función `QuitarTodosEspaciosDeNombres` debe estar definida en el script de declaraciones globales. Una vez definida la función `QuitarTodosEspaciosDeNombres`, la macro se completa y se puede ejecutar.

Nota: en las macros no se pueden usar parámetros ni valores devueltos.

Definir una macro como macro automática

Cuando una macro se define como macro automática, la macro se puede ejecutar automáticamente cuando (i) se inicia Authentic Desktop o (ii) cuando se carga un proyecto de Altova Authentic Desktop en Authentic Desktop. Si quiere que las macros automáticas se ejecuten automáticamente cuando tengan lugar estos dos eventos, marque la casilla *Ejecutar macros automáticas*:

- *Cuando se inicie Authentic Desktop:* marque la casilla *Ejecutar macros automáticas* de la pestaña Script del cuadro de diálogo "Opciones" de Authentic Desktop (**Herramientas | Opciones**).
- *Cuando se cargue un proyecto de Authentic Desktop en Authentic Desktop:* marque la casilla *Ejecutar macros automáticas* en el cuadro de diálogo "Script" (*imagen siguiente*) (**Proyecto | Configuración de script**).



Para definir una macro como macro automática, en el Editor de scripts haga clic con el botón secundario en la macro y seleccione el comando **Establecer como macro automática**. Este comando es un conmutador: haga clic otra vez en el comando para que la macro deje de ser automática.

© 2015 Altova GmbH

1.6.2 Ejecutar una macro

Para ejecutar una macro en el Editor de scripts haga clic con el botón secundario en la macro (en el panel Estructura del proyecto de scripting) y seleccione el comando **Ejecutar macro**.

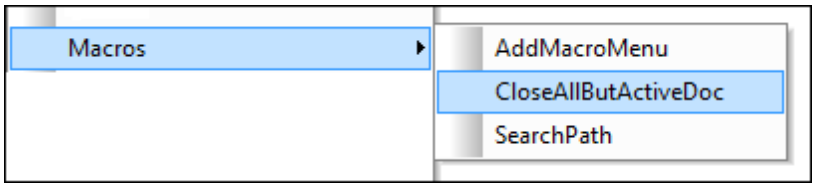
Hay varias maneras de ejecutar una macro desde Authentic Desktop:

- Desde el menú Herramientas | Macros de Authentic Desktop.
- Creando y usando un botón de la barra de herramientas para la macro.
- Creando y usando un comando de menú para la macro.

Recuerde que no se puede ejecutar más de una macro a la vez. Cuando se termina de ejecutar la macro (o el evento), el script se cierra y las variables globales pierden sus valores.

El comando de Authentic Desktop para ejecutar macros

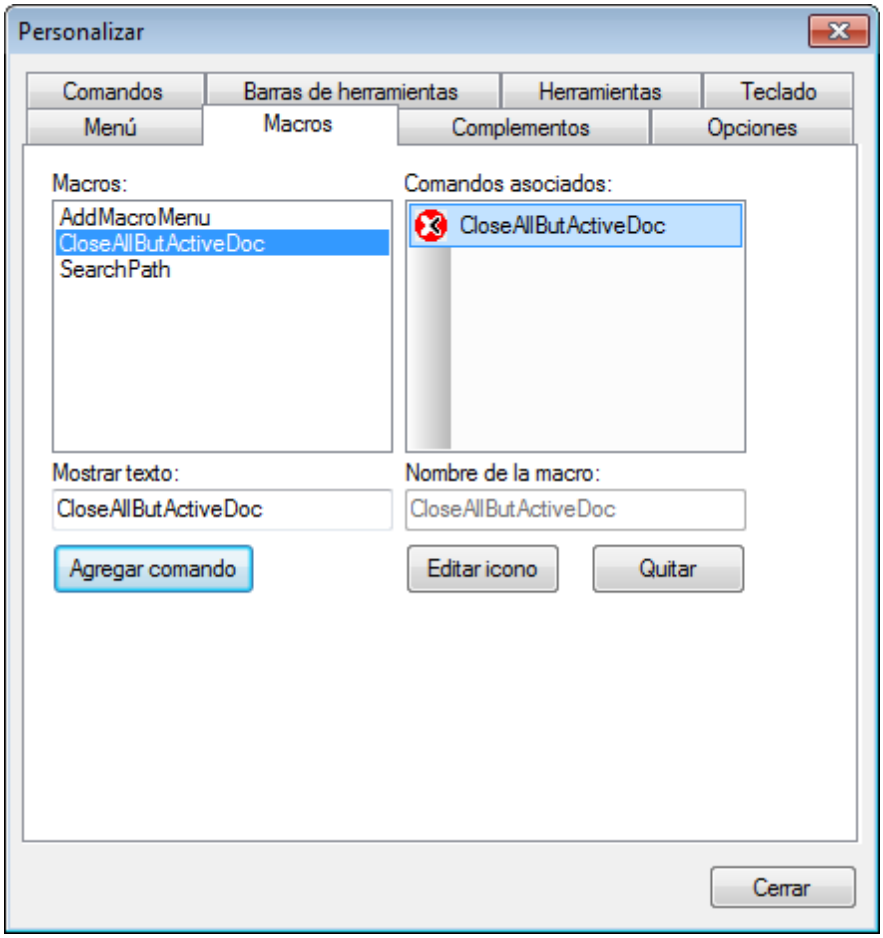
El comando de menú **Herramientas | Macros** (*imagen siguiente*) abre un submenú que enumera todas las macros definidas en el proyecto de scripting que está activo en ese momento en Authentic Desktop. El proyecto de scripting activo es el indicado en la pestaña "Script" del cuadro de diálogo "Opciones" o en el cuadro de diálogo "Scripts".



Ahora seleccione la macro del submenú que desea ejecutar.

Icono de la barra de herramientas

Puede crear un icono de barra de herramientas o un comando de menú que ejecute la macro elegida. Para ello empiece por hacer clic en **Herramientas | Personalizar**. Esto abre el cuadro de diálogo "Personalizar". Haga clic en la pestaña *Macros* (*imagen siguiente*).



Ahora siga estas instrucciones:

1. En la pestaña "Macros" seleccione la macro pertinente en el panel *Macros*. Este panel muestra todas las macros que están disponibles en el proyecto de scripting activo.
2. En el campo de entrada *Mostrar texto* escriba el nombre del icono. Este nombre aparecerá cuando el usuario pase el puntero del mouse sobre el icono.
3. Ahora haga clic en el botón **Agregar comando** para añadir el comando a la lista del panel *Comandos asociados*.
4. Seleccione el comando y haga clic en el botón **Editar icono** para crear un icono nuevo.
5. Arrastre el icono recién creado desde la lista del panel *Comandos asociados* hasta la barra de herramientas o el menú donde desea colocarlo. Cuando el cursor tenga forma de línea o de **I**, suelte el icono.
6. Si quiere, también puede asignar un acceso directo a la macro desde la pestaña *Teclado* del cuadro de diálogo "Personalizar".

Para eliminar el icono de la barra de herramientas o el comando del menú, abra la pestaña *Macros* del cuadro de diálogo "Personalizar" y arrastre el icono desde la barra de herramientas o desde el menú hasta el panel *Comandos asociados*. Ahora seleccione el comando en el panel *Comandos asociados* y haga clic en el botón **Quitar**.

Comando del menú Herramientas

© 2015 Altova GmbH

La API de XMLSpy incluye una función llamada `AddMacroMenuItem()` que sirve para añadir macros a lista del menú **Herramientas | Macros**. Estas son las instrucciones:

1. Añada la macro al menú llamando a la función de la API de XMLSpy `AddMacroMenuItem()`.

```
Application.AddMacroMenuItem("DeleteElements","Delete Elements
Dialog");
```

   - El primer parámetro de la función (`DeleteElements` en el ejemplo anterior) es el nombre de la macro. Si ejecuta la macro y hay un proyecto abierto que tenga scripts asociados, Authentic Desktop primero busca la macro en los scripts del proyecto. Si no hay scripts en el proyecto o si Authentic Desktop no encuentra la macro, entonces se busca la macro en los scripts globales.
   - El segundo parámetro (`Delete Elements Dialog`) es el texto que aparecerá para el elemento de menú.

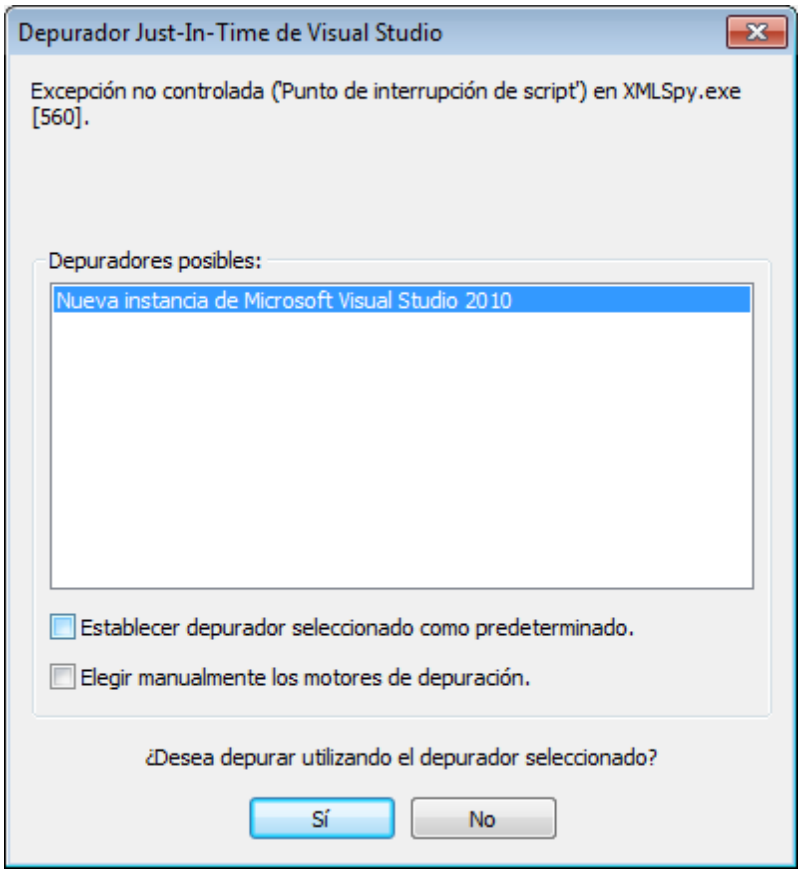
2. Restaure el menú **Herramientas** llamando a `ClearMacroMenu()`. Esto quita todos los elementos de menú añadidos previamente.

La mejor manera de llamar a estas dos funciones es con la macro `Autorun` del proyecto global de scripting o con el evento `On_OpenProject`.

1.6.3 Depurar una macro

Las macros se pueden depurar usando el depurador que tenga instalado en el equipo. Para ello haga clic con el botón secundario en el nombre de la macro (en el panel Estructura del proyecto de scripting) y seleccione el comando **Depurar macro** del menú contextual.

Esto abre el cuadro de diálogo "Depurador Just-In-Time" (*imagen siguiente*), que enumera todos los depuradores que están instalados en el equipo. Seleccione uno de ellos y haga clic en el botón **Sí**.


Depurador Just-In-Time de Visual Studio
Excepción no controlada ('Punto de interrupción de script') en XMLSpy.exe [560].
Depuradores posibles:
Nueva instancia de Microsoft Visual Studio 2010
Establecer depurador seleccionado como predeterminado.
Elegir manualmente los motores de depuración.
¿Desea depurar utilizando el depurador seleccionado?
Sí
No

Entonces se inicia el depurador seleccionado.

© 2015 Altova GmbH

1.7 Notas sobre programación

Es necesario tener en cuenta todos estos puntos:

- En Microsoft .NET Framework se puede acceder por defecto a todos los espacios de nombres y tipos de estos ensamblados .NET:

```
System
System.Data
System.Design
System.Drawing
System.Windows.Forms
System.XML
```

  Al proyecto de scripting se pueden añadir más ensamblados con ayuda del menú contextual del panel Estructura del proyecto del Editor de scripts o en el código fuente de forma dinámica (en tiempo de ejecución) usando `CLR.LoadAssembly`.

- Los parámetros out de los métodos de la API de XMLSpy exigen variables especiales en JScript. Ejemplos:

```
        // usar el método JScript para acceder a los parámetros out
var strError = new Array(1);
var nErrorPos = new Array(1);
var objBadData = new Array(1);
bOK = objDoc.IsValid(strError,nErrorPos,objBadData);END
```

- Los parámetros out de los métodos de .NET Framework exigen variables especiales en JScript. Ejemplos:

```
var dictionary =
CLR.Create( "System.Collections.Generic.Dictionary< System.String,
System.String >" );
dictionary.Add("1", "A");
dictionary.Add("2", "B");

// usar el método JScript para acceder a los parámetros out
var strOut = new Array(1);
if ( dictionary.TryGetValue("1", strOut) ) // TryGetValue
establecerá el parámetro out
    alert( strOut[0] ); // usar el parámetro out
```

- Los métodos de .NET que exigen argumentos enteros no se deberían invocar directamente con objetos Number de JScript que sean valores de punto flotante.

  Por ejemplo, en lugar de

```
var objCustomColor = CLR.Static( "System.Drawing.Color" ). FromArgb( 128,
128, 128 );
```

  recomendamos usar

```
var objCustomColor = CLR.Static( "System.Drawing.Color" ).
FromArgb( Math.floor( 128 ), Math.floor( 128 ), Math.floor( 128 ) );
```

- Para iterar en colecciones .NET puede usar el objeto Enumerator de JScript y el iterador .NET. Ejemplo:

```
// iterar usando el iterador de JScript
var itr = new Enumerator( coll );
for ( ; !itr.atEnd(); itr.moveNext() )
        alert( itr.item() );

// iterar usando el iterador .NET
var itrNET = coll.GetEnumerator();
while( itrNET.MoveNext() )
        alert( itrNET.Current );
```

- Puede crear instancias de las plantillas .NET de esta forma:

```
var coll =
CLR.Create( "System.Collections.Generic.List<System.String>" );
```

o

```
CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary< String, Dictionary< String,
String > >" );
```

- Puede acceder a los valores de .NET Enum de esta forma:

```
        var enumValStretch =
CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch;
```

- Los literales de enumeración, tal y como se definen en las bibliotecas de tipos de Altova, ahora se pueden usar en lugar de valores numéricos.

```
        objExportXMIFileDlg.XMIType = eXMI21ForUML23;
```


© 2015 Altova GmbH

1.7.1 Comandos integrados

En este apartado se explican:

- Comandos integrados
  alert
  conform
  doevents
  CreateForm
  lastform
  prompt
  ShowForm
  watchdog

- Comandos de interoperabilidad de .NET
  CLR.Create
  CLR.Import
  CLR.LoadAssembly
  CLR.ShowImports
  CLR.ShowLoadedAssemblies
  CLR.Static

Comandos integrados

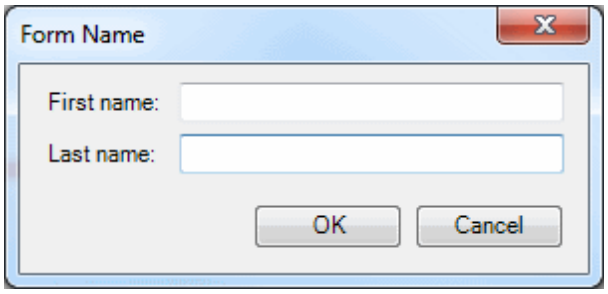
Estos son los comandos integrados disponibles:

ShowForm(strFormName : cadenaDeTexto)
Crea una instancia de un objeto FormularioNuevo a partir del nombre de formulario dado y lo muestra inmediatamente en forma de cuadro de diálogo.
Valor devuelto: un número que representa el DialogResult generado (`System.Windows.Forms.DialogResult`).

Ejemplo:

```
var dialogResult = ShowForm( "FormName" );
```

Muestra el formulario "FormName" como cuadro de diálogo:



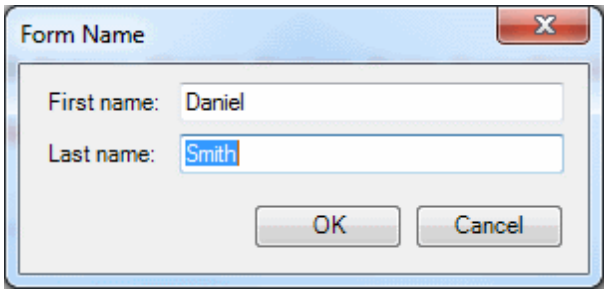
DialogResult se puede evaluar con:

```
if ( dialogResult ==
CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
       alert( "ok" );
else
       alert( "cancel" );
```

CreateForm(strFormName : cadenaDeTexto)
Crea una instancia de un objeto FormularioNuevo a partir del nombre de formulario dado.
Valor devuelto: el objeto Form (`System.Windows.Forms.Form`) con el nombre dado o `null` si existe ningún objeto Form con ese nombre.

```
Ejemplo:
	var myForm = CreateForm( "FormName" );
	if ( myForm != null )
	{
		myForm.textboxFirstName.Text = "Daniel";
		myForm.textboxLastName.Text = "Smith";
		var dialogResult = myForm.ShowDialog();
	}
```

Muestra el formulario "FormName" como cuadro de diálogo. Los cuadros de texto ya están inicializados:



DialogResult se puede evaluar con:

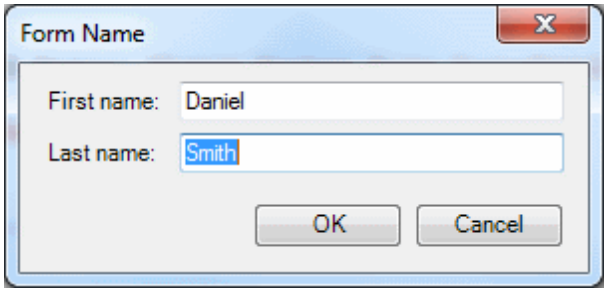
```
if ( dialogResult ==
CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
	alert( "ok" );
else
	alert( "cancel" );
```

lastform
Este campo global se puede usar para acceder fácilmente al último objeto Form que se creó.
Valor devuelto: devuelve una referencia al último objeto Form (`System.Windows.Forms.Form`) que se creó mediante `CreateForm()` o `ShowForm()`.

```
Ejemplo:
	CreateForm( "FormName" );
	if ( lastform != null )
	{
		lastform.textboxFirstName.Text = "Daniel";
		lastform.textboxLastName.Text = "Smith";
		var dialogResult = lastform.ShowDialog();
	}
```

Muestra el formulario "FormName" como cuadro de diálogo. Los cuadros de texto ya están inicializados (como en el ejemplo CreateForm anterior):

© 2015 Altova GmbH



doevents()
Procesa todos los mensajes de Windows que estén en la cola de mensajes.
Valor devuelto: Ninguno

```
Ejemplo:
	for ( i=0; i < nLongLastingProcess; ++i )
	{
		// realizar procesos de larga duración

		doevents(); // procesar mensajes de Windows; dar a la interfaz del
	usuario la oportunidad de actualizarse
	}
```

watchdog(bEnable : booleano)
Los scripts de larga ejecución que consumen mucha CPU hacen que el guardián pregunte al usuario si el script se debe interrumpir. El método guardián `watchdog()` se utiliza para deshabilitar o habilitar este comportamiento. El guardián está habilitado por defecto.
Valor devuelto: Ninguno

```
Ejemplo:
	watchdog( false );   // deshabilitar guardián. Sabemos que la siguiente
	instrucción consume mucha CPU pero finalizará con toda seguridad
	doCPUIntensiveScript();
	watchdog( true );    // volver a habilitar el guardián
```

Consejo práctico:
También puede restaurar el guardián llamando a `watchdog(true)`. Esto puede ser muy práctico antes de ejecutar tareas de larga ejecución (que consumen mucha CPU) para garantizar que tienen asignada la cuota máxima de procesamiento permitida.

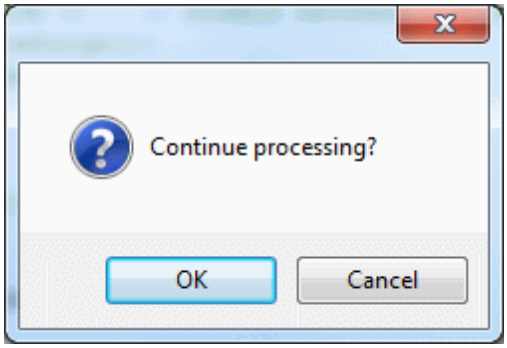
alert(strMessage : cadenaDeTexto) o MsgBox(strMessage : cadenaDeTexto)
Se usa un cuadro de alerta para mostrar un mensaje dado. El usuario debe hacer clic en **Aceptar** para continuar.
Valor devuelto: Ninguno

```
Ejemplo:
	alert( "Hello World" );
```

confirm(strMessage : cadenaDeTexto)
Abre un cuadro de diálogo que muestra el mensaje de confirmación dado. Los cuadros de confirmación se usan a menudo para verificar o aceptar algo. El usuario debe hacer clic en **Aceptar** o **Cancelar** para continuar.
Valor devuelto: un booleano que representa la respuesta del usuario. Si el usuario hace clic en **Aceptar**, el cuadro de diálogo devuelve `true`. Si hace clic en **Cancelar**, el cuadro de diálogo devuelve `false`.

Ejemplo:

```
if ( confirm( "Continue processing?" ) == false )
	return;
```

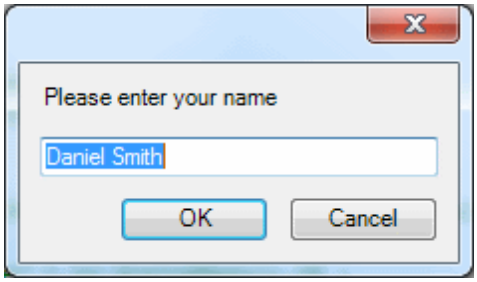


prompt(strMessage : cadenaDeTexto, strDefault : cadenaDeTexto)
Abre un cuadro de diálogo que muestra el mensaje de solicitud dado y un cuadro de texto con una respuesta predeterminada. Los cuadros de solicitud suelen usarse para introducir un valor de cadena sencillo.
Valor devuelto: una cadena que contiene el valor del cuadro de texto o `null` si el usuario hizo clic en **Cancelar**.

Ejemplo:

```
var name = prompt( "Please enter your name", "Daniel Smith" );
if ( name != null )
	alert( "Hello " + name + "!" );
```

 © 2015 Altova GmbH



Comandos de interoperabilidad de .NET

Bajo CLR se ofrecen algunas funciones .NET más para permitir una mayor interoperabilidad.

CLR.Import(strNamespaceCLR : cadenaDeTexto)

Este es el equivalente a la palabra clave *using* de C# e *imports* de VB.Net. Permite obviar los espacios de nombres dados en las llamadas sucesivas, como en `CLR.Create()` y `CLR.Static()`.

Valor devuelto: ninguno

`Ejemplo:`

En lugar de tener que usar siempre nombres completos:

```
if ( ShowForm( "FormName" ) ==
CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
{
	var sName = lastform.textboxFirstName.Text + " " +
lastform.textboxLastName.Text;
	CLR.Static( "System.Windows.Forms.MessageBox" ).Show( "Hello " +
sName );
}
```

se pueden importar los espacios de nombres y usar la forma abreviada:

```
CLR.Import( "System.Windows.Forms" );
if ( ShowForm( "FormName" ) == CLR.Static( "DialogResult" ).OK )
{
	var sName = lastform.textboxFirstName.Text + " " +
lastform.textboxLastName.Text;
	CLR.Static( "MessageBox" ).Show( "Hello " + sName );
}
```

Nota importante:

Al importar un espacio de nombres no se añade ni se carga el ensamblado correspondiente al proyecto de scripting.

Los ensamblados se pueden agregar al proyecto de scripting de forma dinámica (en tiempo de ejecución) en el código fuente con ayuda de `CLR.LoadAssembly`.

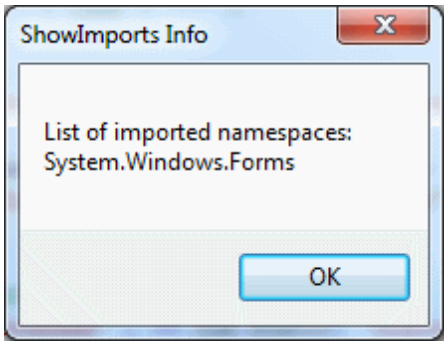
CLR.ShowImports()

Abre un cuadro de diálogo MessageBox que muestra los espacios de nombres importados actualmente. El usuario debe hacer clic en **Aceptar** para continuar.

Valor devuelto: Ninguno

Ejemplo:

CLR.ShowImports();



CLR.LoadAssembly(strAssemblyNameCLR : cadenaDeTexto)

Carga el ensamblado .NET con el nombre de ensamblado dado o con la ruta de acceso de archivo dada.

Valor devuelto: un valor booleano. `True` si se cargó el ensamblado. `False` si no se cargó.

Ejemplo:

```
// establecer el texto del portapapeles (si es posible)
// System.Windows.Clipboard es parte del ensamblado PresentationCore, así
que queremos cargar este ensamblado primero:
if ( CLR.LoadAssembly( "PresentationCore, Version=3.0.0.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35", true ) )
{
	var clipboard = CLR.Static( "System.Windows.Clipboard" );
	if ( clipboard != null )
		clipboard.SetText( "HelloClipboard" );
}
```

CLR.ShowLoadedAssemblies()

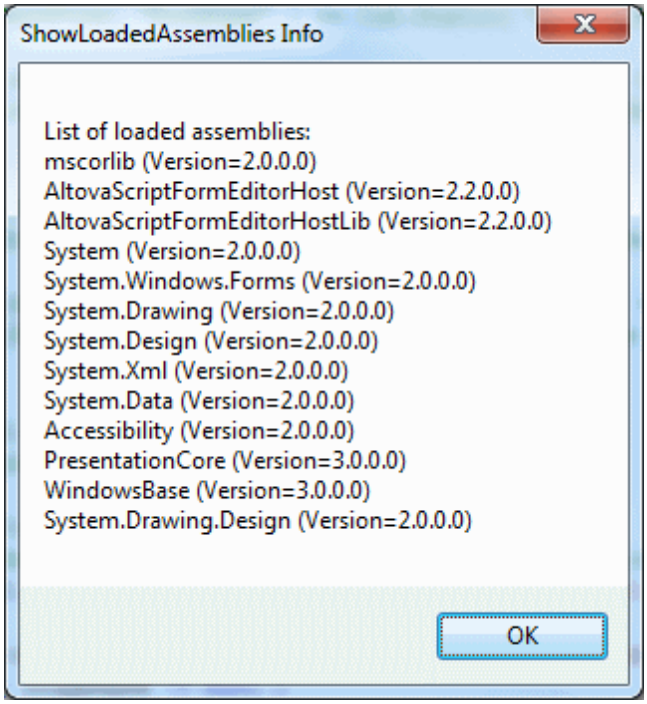
Abre un cuadro de diálogo MessageBox que enumera los ensamblados que están cargados actualmente. El usuario debe hacer clic en **Aceptar** para continuar.

Valor devuelto: Ninguno

Ejemplo:

```
CLR.ShowLoadedAssemblies();
```


© 2015 Altova GmbH



CLR.Create(strTypeNameCLR : cadenaDeTexto, argumentos del constructor ...)
Crea una nueva instancia de objeto .NET para el typeName dado. Si se pasa más de un argumento los argumentos sucesivos se interpretan como argumentos para el constructor del objeto .NET.
Valor devuelto: una referencia al objeto .NET creado

Ejemplos:

```
var objArray = CLR.Create("System.Collections.ArrayList");

var newItem = CLR.Create( "System.Windows.Forms.ListViewItem",
"NewItemText" );

var coll =
CLR.Create( "System.Collections.Generic.List<System.String>" );

CLR.Import( "System" );
CLR.Import( "System.Collections.Generic" );
var dictionary = CLR.Create( "Dictionary< String, Dictionary< String,
String > >" );
```

CLR.Static(strTypeNameCLR : cadenaDeTexto)
Da acceso a los tipos de .NET que no tienen instancias y que solo contienen miembros estáticos.
Valor devuelto: una referencia al objeto .NET estático

Ejemplo:

```
var enumValStretch =
CLR.Static( "System.Windows.Forms.ImageLayout" ).Stretch
```

```
var clipboard = CLR.Static( "System.Windows.Clipboard" );
clipboard.SetText( "HelloClipboard" );

if ( ShowForm( "FormName" ) ==
CLR.Static( "System.Windows.Forms.DialogResult" ).OK )
       alert( "ok" );
else
       alert( "cancel" );
```

Uso de los formularios y comandos

Este es el uso de los formularios:

Con los objetos Form se puede acceder a la estructura de componentes mediante acceso de campo:

Por ejemplo, imagine que hay designado este objeto Form:

```
MiFormulario
       ButtonPanel
              OkButton
              CancelButton
       TextEditor
       AxMediaPlayer1

TrayComponents:
       MiTemporizador
```

Desde el script se puede crear una instancia del objeto Form:

```
var objForm = CreateForm("MiFormulario");
```

Para acceder a uno de sus componentes puede usar el acceso de campo:

```
objForm.ButtonPanel.OkButton.Enabled = false;
```

o

```
objForm.TextEditor.Text = "Hola mundo";
```

Para acceder a Tray Components use este método en el objeto Form:

```
var objTrayComponent = <A form object>.GetTrayComponent(strComponentName :
String);
```

Para obtener una referencia al componente Timer y habilitarlo puede utilizar:

```
var objTimer = objForm.GetTrayComponent("MiTemporizador");
objTimer.Enabled = true;
```

Para los controles ActiveX puede acceder al objeto COM subyacente mediante la propiedad OCX:

```
var ocx = lastform.AxMediaPlayer1.OCX; // obtener objeto COM subyacente
ocx.enableContextMenu = true;
```


© 2015 Altova GmbH

```
ocx.URL = "mms://apasf.apa.at/fm4_live_worldwide";
```

1.8 Migración al editor de scripts versión 2010 y superior

El Editor de scripts de Authentic Desktop versión 2010 y superior está basado en una tecnología subyacente distinta a las de las versiones previas del Editor. Por tanto, quizás deba modificar los proyectos de scripting creados con versiones de Authentic Desktop anteriores a la versión 2010. A este respecto debe tener en cuenta que:

- Si un proyecto de scripting antiguo (archivo `.prj`) se abre en el nuevo Editor de scripts (versión 2010 o superior), la disposición gráfica de los formularios se migrará lo más fielmente posible y los scripts se copiarán tal y como están en el archivo `.prj`. Después tendrá que modificar los scripts para ajustarlos a la nueva tecnología utilizada por el Editor de scripts versión 2010 o superior, que se describe en esta documentación.
- El objeto `TheView`: el antiguo entorno de scripting ofrecía una propiedad artificial llamada `TheView` a la que solo se podía acceder desde controladores de eventos. Se usaba para acceder al formulario que desencadenó el evento (bien directamente o desde uno de sus controles secundarios). El nuevo entorno de scripting **no ofrece esta propiedad** artificial, pero sigue ofreciendo la misma función gracias a las funciones auxiliares de scripting integradas ortogonales y a la potencia de .NET Framework.
- Como todos los controladores de eventos del nuevo entorno de scripting obtienen un objeto sender como primer parámetro, el código que desencadenó el evento siempre está disponible. Al llamar a la función .NET `FindForm()` en el objeto sender, podemos acceder al objeto Form con facilidad. Otra opción (si solamente hay un objeto Form) es usar la propiedad integrada `lastform`. Recuerde que el uso de `lastform` no se limita a los controladores de eventos (como ocurría con `TheView`), sino que se puede utilizar en cualquier parte del código de script.

A continuación puede ver la lista de métodos y propiedades del objeto `TheView` acompañados de los mecanismos equivalentes ofrecidos por el entorno de scripting de la versión 2010 y superior.

Métodos

Estos métodos que ofrecía el objeto `TheView` deben migrarse tal y como explicamos a continuación:

`Cancel()`

En el nuevo entorno de scripting puede conseguir lo mismo con: `lastform.Close(); // Use .NET Form.Close()`

`IsFormOpen(Name as String) as Boolean`

Como en los formularios .NET no es lo mismo mostrar un formulario que crear una instancia del formulario, este concepto no tiene un equivalente directo. En cambio, el usuario puede preguntar si se está mostrando cierto formulario. Por ejemplo:

```
var objFormPencilSelector = CreateForm("PencilSelector");
var objFormColorSelector = CreateForm("ColorSelector");
       ...
// Anywhere in code ...

if(objFormColorSelector.Visible)
{
       ...
}
```


© 2015 Altova GmbH

FormFind(Name as String) as Object
El nuevo entorno de scripting permite crear instancias de más formularios del mismo tipo. En el entorno de scripting antiguo cada formulario podía existir una vez como máximo (como Singleton). Por tanto, en el nuevo entorno de scripting no hay un equivalente de `FormFind()`.

OpenDoc(File as String)
Puede conseguir lo mismo con: `Application.OpenDocument( File as String )`

PumpData()
Esto equivale a la función integrada `doevents()` que procesa todos los mensajes de Windows que están en ese momento en la cola de mensajes.

RunClick(), RunInitialize(), RunTerminate()
No hay un equivalente directo para estos métodos. La alternativa es llamar directamente a los controladores correspondientes.

Propiedades

Estas propiedades que ofrecía el objeto `TheView` deben migrarse tal y como explicamos a continuación:

ToolTipText as String
Para usar información rápida en el nuevo entorno de scripting, puede usar la infraestructura .NET. Así podrá controlar el comportamiento de la información rápida (ajustando retrasos, cuándo aparece, etc.). Por ejemplo, si quiere ofrecer información rápida para un formulario compuesto por dos controles, puede añadir este código al controlador de eventos `Load` del formulario:

```
              //Aparece cada vez que el usuario carga el formulario.
              function MyForm_Load( objSender, e_EventArgs )
              {
                     // Crear la información rápida ToolTip y asociarla al
contenedor del formulario.
                     var toolTip = CLR.Create("System.Windows.Forms.ToolTip");

                     // Establecer los retrasos para la información rápida.
                     toolTip.AutoPopDelay = 3000;
                     toolTip.InitialDelay = 1000;
                     toolTip.ReshowDelay = 500;

                     // Obligar a que aparezca el texto de información rápida
                     // independientemente de si el formulario está activo o no.
                     toolTip.ShowAlways = true;

                     // Establecer el texto de información rápida para varios
controles.
                     toolTip.SetToolTip(objSender.ProgressBar1,
                                   "Shows the progress of the operation");
                     toolTip.SetToolTip(objSender.Button1,
                                   "Click Button to start the processing");
              }
```

Color as Long

Como todos los controles y formularios del nuevo entorno de scripting son controles .NET del espacio de nombres `System.Windows.Forms`, hay muchas posibilidades para modificar colores, la imagen de fondo, las fuentes y otros aspectos visuales. Por ejemplo, todos los componentes visuales tienen las propiedades `BackColor` y `ForeColor` que permiten modificar el aspecto visual del componente. Por ejemplo, puede usar este controlador para cambiar el color de un botón en tiempo de ejecución:

```
	function TestForm_Button1_Click( objSender, e_EventArgs )
	{
		objSender.BackColor =
CLR.Static( "System.Drawing.Color" ).SlateBlue;
	}
```

Para más información puede consultar la documentación de .NET sobre este tema:
http://msdn.microsoft.com/es-es/library/system.windows.forms.aspx

© 2015 Altova GmbH

2 Complementos de Authentic Desktop para entornos IDE

Authentic Desktop permite crear complementos para entornos IDE e integrarlos en Authentic Desktop.

Puede usar estos complementos para:

- Configurar su versión de Authentic Desktop y agregar comandos a través de menús, iconos, botones, etc.
- Reaccionar a eventos desde Authentic Desktop.
- Ejecutar su propio código dentro de Authentic Desktop con acceso a toda la API de Authentic Desktop

Authentic Desktop espera que el complemento implemente la interfaz IXMLSpyPlugIn. Con el paquete de instalación vienen ejemplos en VB.NET, C# y C++, que se instalan en la carpeta de la carpeta de instalación y que se localizan en la carpeta `Authentic2016\AuthenticExamples\IDEPlugin` Authentic Desktop de su instalación.

Windows XP	`C:/Documents and Settings/<usuario>/Mis Documentos`
Windows Vista, Windows 7/8	`C:/Users/<usuario>/Documentos`

Consulte el apartado *Archivos de muestra ATL* para ver un ejemplo con C++.

2.1 Cómo registrar complementos para entornos IDE

Authentic Desktop mantiene una clave en el registro en la que almacena todos los complementos registrados para entornos IDE:

```
HKEY_CURRENT_USER\Software\Altova\XML Spy\PlugIns
```

Todos los valores de esta clave se tratan como referencias a los complementos registrados y deben tener este formato:

Nombre del valor:	`ProgID` del complemento
Tipo de valor:	debe ser `REG_SZ`
Datos del valor:	`CLSID` del componente

Cada vez que se inicia la aplicación, se examinan los valores de la clave `PlugIns` y se cargan los complementos que estén registrados.

Registrar el complemento a mano
Para registrar un complemento a mano utilice el cuadro de diálogo "Personalizar" del menú **Herramientas** de Authentic Desktop. Con el botón **Agregar complemento...** puede especificar qué DLL implementa el complemento. Authentic Desktop registra el DLL como servidor de COM y añade la correspondiente entrada en su clave `PlugIns`.

Si tiene problemas durante el registro manual, compruebe si el CLSID del complemento coincide con el de la clave `PlugIns`. Si es así, puede que el nombre del DLL del complemento no fuera lo suficientemente único. Utilice otro nombre o registre el complemento directamente (ver más abajo).

Registrar el complemento directamente
Los complementos se pueden registrar directamente: para ello debe registrar el DLL primero y después agregar el valor que le corresponda en la clave `PlugIns` de Authentic Desktop durante la configuración del complemento, por ejemplo. El nuevo complemento se activará la próxima vez que se inicie Authentic Desktop.

Crear complementos
El código origen para los complementos de prueba se han proporcionado en (My) Documents folder: `Examples\IDEPlugin` folder. Para generar un complemento desde un código fuente siga los siguientes pasos:

1. Abra la solución que desea generar como complemento en Visual Studio.
2. Genere el complemento con el comando del menú Crear.
3. El archivo DLL del complemento se creará en la carpeta `Bin` o `Debug`. Este archivo DLL es el archivo que debe agregarse como complemento (consulte más arriba).

© 2015 Altova GmbH

2.2 Controles ActiveX

Los controles ActiveX son compatibles y los complementos para entornos IDE que también sean control ActiveX aparecerán en una barra de control de diálogo. Además, en la carpeta `Examples\XMLSpyPlugInActiveX` de la carpeta de la aplicación hay un complemento de muestra que también es un control ActiveX.

2.3 Configuración XML

El complemento para entornos IDE permite modificar la interfaz de usuario de Authentic Desktop. Esto se hace describiendo cada una de las modificaciones por separado mediante secuencias de datos XML. La configuración XML se pasa a Authentic Desktop con el método GetUIModifications de la interfaz IXMLSpyPlugIn.

El archivo XML que contiene las modificaciones de la interfaz de usuario para el complemento debe tener esta estructura:

```
<ConfigurationData>
  <ImageFile>ruta de acceso del archivo de imagen</ImageFile>
  <Modifications>
   <Modification>
    ...
   </Modification>
   ...
  </Modifications>
 </ConfigurationData>
```

Puede definir iconos o botones de barras de herramientas para los nuevos comandos de menú que el complemento añade a la interfaz gráfica de Authentic Desktop. La ruta de acceso del archivo que contiene las imágenes se establece con el elemento ImageFile. Cada imagen debe tener 16 x 16 píxels y usar 256 colores como máximo. Las referencias a la imagen deben organizarse de izquierda a derecha en una sola línea (<ImageFile>...). El valor de índice de la imagen situada más a la derecha es cero.

El elemento Modifications puede tener un número indefinido de elemento secundarios Modification. Cada elemento Modification define un cambio concreto en la interfaz gráfica estándar de Authentic Desktop. A partir de la versión 4.3 también se pueden quitar elementos de la interfaz de Authentic Desktop.

Estructura de los elementos Modification
Todos los elementos Modification están compuestos por dos elementos secundarios:

```
<Modification>
 <Action>tipo de acción</Action>
 <UIElement Type="tipo de elemento de interfaz gráfica">
 </UIElement>
</Modification>
```

Estos son los valores válidos para el elemento Action:

Add - para agregar el elemento a la interfaz de Authentic Desktop
Hide - para ocultar el elemento en la interfaz de Authentic Desktop
Remove - para quitar el elemento del cuadro de lista *Comandos* del cuadro de diálogo "Personalizar".

También puede combinar varios valores en el elemento Action (p. ej. Hide Remove).

El elemento UIElement describe los elementos de la interfaz gráfica de Authentic Desktop, nuevos o ya existentes. Se trata de estos elementos: barras de herramientas nuevas, botones

© 2015 Altova GmbH

nuevos, menús o comandos de menús. El atributo **type** define qué elemento es descrito por el elemento XML.

Elementos secundarios frecuentes de UIElement

Los elementos ID y Name son válidos para todos los tipos diferentes de fragmentos XML UIElement. Sin embargo, se puede ignorar uno de los valores para un tipo concreto de UIElement (p. ej. Name se ignora para un separador).

```
<ID></ID>
<Name></Name>
```

Si UIElement describe un elemento actual de la interfaz gráfica, el valor del elemento ID viene predefinido en Authentic Desktop. Lo normal es que estos valores del elemento ID no sean conocidos. Si el fragmento XML describe una parte nueva de la interfaz gráfica, entonces el ID es aleatorio y su valor debería ser menor a 1000.

El elemento Name establece el valor textual. Los elementos actuales de la interfaz gráfica se pueden identificar con su nombre (p. ej. menús y elementos de menús). En el caso de elementos nuevos de la interfaz, el elemento Name establece el título (p. ej. el título de una barra de herramientas o el texto de un comando de menú).

Barras de herramientas y menús

Para definir una barra de herramienta hace falta especificar el ID y el nombre de la barra de herramientas. Para las barras de herramientas ya existentes se puede indicar el nombre solamente o el ID, si se conoce. Para crear una barra de herramientas nueva es necesario indicar ambos valores. El atributo **type** debe ser igual a "ToolBar".

```
<UIElement Type="ToolBar">
 <ID>1</ID>
 <Name>TestPlugIn</Name>
</UIElement>
```

Para especificar un menú de Authentic Desktop son necesarios dos parámetros:

- El ID de la barra de menú que contiene el menú. Si en la ventana principal no hay ningún documento XML abierto, el ID de la barra de menú es 128. Si hay algún documento XML abierto, el ID de la barra de menú es 129.
- El nombre del menú. Los menús no tienen un valor ID asociado. En el fragmento siguiente, por ejemplo, se define el menú Edición de la barra de menú que está activa cuando hay un documento XML abierto como mínimo:

```
<UIElement Type="Menu">
 <ID>129</ID>
 <Name>Edición</Name>
</UIElement>
```

Si quiere crear un menú nuevo hace falta usar un elemento más: el elemento Place, que define la posición del nuevo menú en la barra de menús:

```
<UIElement Type="Menu">
 <ID>129</ID>
 <Name>Menú PlugIn</Name>
```

```
  <Place>12</Place>
 </UIElement>
```

El valor -1 para el elemento Place coloca el botón o menú nuevo al final de la barra de herramientas o del menú respectivamente.

Comandos

Si añade un comando nuevo, a través de una barra de herramientas o de un comando de menú, el fragmento UIElement puede contener cualquiera de estos subelementos:

```
 <MacroName></MacroName>
 <Info></Info>
 <ImageID></ImageID>
```

Si se especifica el elemento MacroName, Authentic Desktop busca una macro que tenga el mismo nombre en el entorno de scripting y la ejecuta cada vez que se procese este comando.

El elemento Info contiene una breve descripción que aparece en la barra de estado cuando se pasa el puntero del mouse por encima del comando asociado (tanto por el botón como por el comando de menú).

ImageID define el índice del icono del archivo de imagen externo. Recuerde que todos los iconos se almacenan en un solo archivo de imagen.

Para definir un botón de barra de herramientas debe crear un elemento UIElement con esta estructura:

```
 <UIElement Type="ToolBarItem">
  <!--no volver a utilizar ID locales incluso si los comandos hacen lo mismo-->
  <ID>5</ID>
  <Name>Abrir archivo del repositorio...</Name>
  <!--Establecer posición en -1 si se trata del primer botón que se debe
insertar-->
  <Place>-1</Place>
  <ImageID>0</ImageID>
  <ToolBarID>1</ToolBarID>
  <!--en lugar del ID de la barra de herramientas también se puede usar el
nombre de la barra de herramientas-->
  <ToolBarName>TestPlugIn</ToolBarName>
 </UIElement>
```

Otros elementos que se pueden utilizar para declarar un botón de barra de herramientas son Place, ToolBarID y ToolBarName. Los elementos ToolBarID y ToolBarName sirven para identificar la barra de herramientas que contiene el botón nuevo o el botón actual. El valor textual de ToolBarName distingue entre mayúsculas y minúsculas. El atributo **type** (del elemento UIElement) debe ser igual a "ToolBarItem".

Para definir un comando de menú, además de los elementos estándar, también están los elementos MenuID, Place y Parent. El elemento MenuID puede ser 128 o 129. Para más información sobre estos valores consulte el párrafo anterior *Barras de herramientas y menús*.

El elemento Parent sirve para identificar el **menú** donde se debe insertar el comando de menú

 © 2015 Altova GmbH

nuevo. Como los comandos de los submenús no tienen un ID único de Windows, necesitamos identificar de otra manera el primario del comando de menú. El valor del elemento Parent es una ruta de acceso al comando de menú.

El valor textual del elemento Parent debe equivaler al nombre del **menú primario del submenú** y el nombre del submenú debe ir separado por dos puntos. Si el menú no es un primario (porque no es un submenú) añada dos puntos al principio del nombre. El atributo **type** debe tener el valor "MenuItem". Por ejemplo, este elemento UIElement define un comando de menú:

```
<UIElement Type="MenuItem">
 <!--este elemento es un ID de comando local-->
 <ID>3</ID>
 <Name>Abrir archivo de repositorio...</Name>
 <Place>-1</Place>
 <MenuID>129</MenuID>
 <Parent>:Menú PlugIn</Parent>
 <ImageID>0</ImageID>
</UIElement>
```

Authentic Desktop permite añadir separadores de barras de herramientas y de menús si el valor del elemento ID es 0.

2.4 Archivos de muestra ATL

Esta sección explica cómo crear un sencillo DLL para el complemento de Authentic Desktop para entornos IDE usando la biblioteca ATL. Para generar el DLL es necesario saber trabajar con la biblioteca ATL, conocer los asistentes que generan objetos ATL nuevos y tener conocimientos de MS Visual Studio.

Para acceder a la API la implementación importa la biblioteca de tipos de Authentic Desktop. El código lee varias propiedades y llama a métodos usando los punteros inteligentes que ofrece la instrucción #import.

Además el código de muestra usa la clase MFC CString y macros de conversión ATL como W2T.

En general, estos son los pasos necesarios para crear un DLL con la biblioteca ATL:

1. Abra VisualStudio y seleccione **Nuevo** en el menú **Archivo**.
2. Seleccione la pestaña *Proyectos*.
3. Seleccione *Asistente para proyectos ATL* y escriba el nombre del proyecto.
4. Si quiere usar clases MFC o quiere crear un proyecto para el código de muestra, seleccione *Compatibilidad para MFC*.

   Una vez creados los archivos de proyecto puede añadir un objeto ATL para implementar la interfaz IXMLSpyPlugIn:

1. En el menú **Insertar** seleccione **Objeto ATL nuevo**.
2. En el asistente seleccione *Objeto simple* y haga clic en **Siguiente**.
3. Escriba un nombre para el objeto.
4. En la pestaña *Atributos* seleccione *Personal* para el tipo de interfaz y deshabilite la opción *Agregación*.

El resultado es un código base para la implementación de la interfaz del complemento para entornos IDE. Consulte el resto de apartados de esta sección para modificar el código y añadir funciones básicas.

© 2015 Altova GmbH

2.4.1 Descripción de interfaz (IDL)

El IDL del objeto ATL recién creado contiene una declaración para una interfaz COM.

- Esta declaración de interfaz debe reemplazarse con la declaración de IXMLSpyPlugIn tal y como aparece más abajo.
- El IDL también debe incluir la definición de la enumeración SPYUpdateAction.
- Reemplace el nombre de interfaz predeterminado generado (creado con el asistente) con el nombre *IXMLSpyPlugIn* en la declaración coclass. El IDL se debería parecer al código del ejemplo siguiente:

Tras crear el objeto ATL, es necesario implementar la interfaz del complemento de Authentic Desktop para entornos IDE:

```
import "oaidl.idl";
import "ocidl.idl";

// ----- inserte este bloque en su archivo IDL -----
 typedef enum {
  spyEnable = 1,
  spyDisable = 2,
  spyCheck = 4,
  spyUncheck = 8
 } SPYUpdateAction;

// ----- fin del bloque insertado ----


// ----- Por, ej. entrada de interfaz generada automáticamente por el asistente
ATL -----
// [
//   object,
//   uuid(AB7CD86A-8145-429A-A1F3-270692EO8AFC),

//   helpstring("IXMLSpyPlugIn Interface")
//   pointer_default(unique)
// ]
// interface IXMLSpyPlugIn : IUnknown
// {
// };

// ----- fin de la entrada de interfaz generada automáticamente


// ----- reemplazar la entrada de interfaz (ver más arriba) generada por el
asistente ATL con este bloque -----

 [
  odl,
  uuid(88F2A622-4B7E-42CD-8D04-3C0E5389DD85),
  helpstring("IXMLSpyPlugIn Interface")
 ]
 interface IXMLSpyPlugIn : IUnknown
  {
```

```
   HRESULT _stdcall OnCommand([in] long nID, [in] IDispatch* pXMLSpy);

   HRESULT _stdcall OnUpdateCommand([in] long nID, [in] IDispatch* pXMLSpy,
[out, retval] SPYUpdateAction* pAction);

   HRESULT _stdcall OnEvent([in] long nEventID, [in] SAFEARRAY(VARIANT)*
arrayParameters, [in] IDispatch* pXMLSpy, [out, retval] VARIANT* pReturnValue);

   HRESULT _stdcall GetUIModifications([out, retval] BSTR* pModificationsXML);

   HRESULT _stdcall GetDescription([out, retval] BSTR* pDescription);
  };

// ----- fin del bloque -----


// ----- El código que aparece ahora lo genera automáticamente el asistente ATL
y puede ser diferente en cada caso -----

 [
 uuid(24FE0D1B-3FC0-494E-B36E-1D4CE412B014),
 version(1.0),
 helpstring("XMLSpyIDEPlugInDLL 1.0 Type Library")
 ]
 library XMLSPYIDEPLUGINDLLLib
 {
 importlib("stdole32.tlb");
 importlib("stdole2.tlb");

 [
  uuid(3800E791-7F6B-4ACD-9E32-2AC184444501),
  helpstring("XMLSpyIDEPlugIn Class")
 ]
 coclass XMLSpyIDEPlugIn
 {
  [default] interface IXMLSpyPlugIn;   // ----- definir IXMLSpyPlugIn como
interfaz predeterminada -----
 };
};
```


© 2015 Altova GmbH

2.4.2 Definición de clase

En la definición de clase del objeto ATL deben realizarse varios cambios. La clase debe derivarse de IXMLSpyPlugIn, el *mapa de la interfaz* necesita una entrada para IXMLSpyPlugIn y los métodos de la interfaz del complemento para entornos IDE debe declararse:

```
#ifndef __XMLSPYIDEPLUGIN_H_
#define __XMLSPYIDEPLUGIN_H_

#include "resource.h"       // símbolos principales

/////////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn
class ATL_NO_VTABLE CXMLSpyIDEPlugIn :
 public CComObjectRootEx<CComSingleThreadModel>,
 public CComCoClass<CXMLSpyIDEPlugIn, &CLSID_XMLSpyIDEPlugIn>,
 public IXMLSpyPlugIn
{
public:
 CXMLSpyIDEPlugIn()
 {
 }

DECLARE_REGISTRY_RESOURCEID(IDR_XMLSPYIDEPLUGIN)
DECLARE_NOT_AGGREGATABLE(CXMLSpyIDEPlugIn)

DECLARE_PROTECT_FINAL_CONSTRUCT()

BEGIN_COM_MAP(CXMLSpyIDEPlugIn)
 COM_INTERFACE_ENTRY(IXMLSpyPlugIn)
END_COM_MAP()

// IXMLSpyIDEPlugIn
public:
 virtual HRESULT _stdcall OnCommand(long nID, IDispatch* pXMLSpy);

 virtual HRESULT _stdcall OnUpdateCommand(long nID, IDispatch* pXMLSpy,
SPYUpdateAction* pAction);

 virtual HRESULT _stdcall OnEvent(long nEventID, SAFEARRAY **arrayParameters,
IDispatch* pXMLSpy, VARIANT* pReturnValue);

 virtual HRESULT _stdcall GetUIModifications(BSTR* pModificationsXML);

 virtual HRESULT _stdcall GetDescription(BSTR* pDescription);
};

#endif //__XMLSPYIDEPLUGIN_H_
```

2.4.3 Implementación

El código de este apartado muestra una sencilla implementación de un complemento de Authentic Desktop para entornos IDE. Añade un comando de menú y un separador (disponible con Authentic Desktop) al menú **Herramientas**. Dentro del método OnUpdateCommand() el comando nuevo solamente se habilita cuando el documento activo se abre en la vista **Cuadrícula**. El comando busca el elemento XML que está resaltado y abre cualquier dirección URL empezando con "http://" a partir del valor textual del elemento.

```
////////////////////////////////////////////////////////////////////////////
// CXMLSpyIDEPlugIn

#import "XMLSpy.tlb"
using namespace XMLSpyLib;

HRESULT CXMLSpyIDEPlugIn::OnCommand(long nID, IDispatch* pXMLSpy)
{
 USES_CONVERSION;

 if(nID == 1) {
  IApplicationPtripSpyApp;

  if(pXMLSpy) {
   if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)
&ipSpyApp))) {
    IDocumentPtr ipDocPtr = ipSpyApp->ActiveDocument;

    // imaginamos que la vista Cuadrícula está activa
    if(ipDocPtr) {
     IGridViewPtr ipGridPtr = ipDocPtr->GridView;

     if(ipGridPtr){
      IXMLDataPtr  ipXMLData = ipGridPtr->CurrentFocus;

      CString strValue = W2T(ipXMLData->TextValue);

      if(!strValue.IsEmpty() && (strValue.Left(7) == _T("http://")))
       ::ShellExecute(NULL,_T("open"),W2T(ipXMLData-
>TextValue),NULL,NULL,SW_SHOWNORMAL);
     }
    }
   }
  }
 }

 return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnUpdateCommand(long nID, IDispatch* pXMLSpy,
SPYUpdateAction* pAction)
{
 *pAction = spyDisable;
```


© 2015 Altova GmbH

```
 if(nID == 1) {
  IApplicationPtripSpyApp;

  if(pXMLSpy) {
   if(SUCCEEDED(pXMLSpy->QueryInterface(__uuidof(IApplication),(void **)
&ipSpyApp))) {
    IDocumentPtr ipDocPtr = ipSpyApp->ActiveDocument;

    // habilitar solamente si la vista Cuadrícula está activa
    if((ipDocPtr != NULL) && (ipDocPtr->CurrentViewMode == spyViewGrid))
     *pAction = spyEnable;
   }
  }
 }

 return S_OK;
}


HRESULT CXMLSpyIDEPlugIn::OnEvent(long nEventID, SAFEARRAY **arrayParameters,
IDispatch* pXMLSpy, VARIANT* pReturnValue)
{
 return S_OK;
}

HRESULT CXMLSpyIDEPlugIn::GetUIModifications(BSTR* pModificationsXML)
{
 CComBSTR bstrMods = _T(" \
    <ConfigurationData>\
     <Modifications>");
 // agregar "Abrir URL..." al menú Herramientas
 bstrMods.Append (_T(" \
      <Modification> \
       <Action>Add</Action> \
       <UIElement type=\"MenuItem\"> \
        <ID>1</ID> \
        <Name>Abrir URL...</Name> \
        <Place>0</Place> \
        <MenuID>129</MenuID> \
        <Parent>:Herramientas</Parent> \
       </UIElement> \
      </Modification>"));
 // agregar separador al menú Herramientas
 bstrMods.Append (_T(" \
      <Modification> \
       <Action>Add</Action> \
       <UIElement type=\"MenuItem\"> \
        <ID>0</ID> \
        <Place>1</Place> \
        <MenuID>129</MenuID> \
        <Parent>:Herramientas</Parent> \
       </UIElement> \
```

```
      </Modification>"));
 // finalizar descripción de la modificación
 bstrMods.Append (_T(" \
     </Modifications> \
    </ConfigurationData>"));

 return bstrMods.CopyTo(pModificationsXML);
}


HRESULT CXMLSpyIDEPlugIn::GetDescription(BSTR* pDescription)
{
 CComBSTR bstrDescr = _T("ATL C++ XMLSpy IDE PlugIn;Este complemento demuestra la
implementación de un DLL ATL sencillo como complemento de XMLSpy para entornos
IDE.");
 return bstrDescr.CopyTo(pDescription);
}
```


© 2015 Altova GmbH

2.5 IXMLSpyPlugIn

Métodos

OnCommand
OnUpdateCommand
OnEvent
GetUIModifications
GetDescription

Descripción

Si se añade un DLL a Authentic Desktop como complemento para entornos IDE, debe registrar un componente COM que responda a una interfaz IXMLSpyPlugIn con el uuid(88F2A622-4B7E-42CD-8D04-3C0E5389DD85) reservado. De lo contrario no se reconocerá que es un complemento.

2.5.1 OnCommand

Declaración: `OnCommand(nID as long, pXMLSpy as IDispatch)`

Descripción

Al método `OnCommand()` de la implementación de la interfaz se le llama cada vez que se procesa un comando añadido por el complemento, ya sea un comando de menú o un botón de barra de herramientas. `nID` almacena el ID del comando definido por el elemento `ID` del correspondiente elemento `UIElement`.

`pXMLSpy` almacena una referencia a la interfaz de envío del objeto `Application` de Authentic Desktop.

Ejemplo

```
Public Sub IXMLSpyPlugIn_OnCommand(ByVal nID As Long, ByVal pXMLSpy As Object)
    If (Not (pXMLSpy Is Nothing)) Then
        Dim objDlg
        Dim objDoc As XMLSpyLib.Document
        Dim objSpy As XMLSpyLib.Application
        Set objSpy = pXMLSpy

        If nID = 3 Or nID = 5 Then
            Set objDlg = CreateObject("MSComDlg.CommonDialog")
            objDlg.Filter = "XML Files (*.xml)|*.xml|All Files (*.*)|*.*||"
            objDlg.FilterIndex = 1
            objDlg.ShowOpen

            If Len(objDlg.FileName) > 0 Then
                Set objDoc = objSpy.Documents.OpenFile(objDlg.FileName, False)
                Set objDoc = Nothing
            End If
        End If

        If nID = 4 Or nID = 6 Then
            Set objDlg = CreateObject("MSComDlg.CommonDialog")
            objDlg.Filter = "All Files (*.*)|*.*||"
            objDlg.Flags = cdlOFNPathMustExist
            objDlg.ShowSave

            If Len(objDlg.FileName) > 0 Then
                Set objDoc = objSpy.ActiveDocument

                If Not (objDoc Is Nothing) Then
                    objDoc.SetPathName objDlg.FileName
                    objDoc.Save
                    Set objDoc = Nothing
                End If
            End If
        End If

        Set objSpy = Nothing
    End If
End Sub
```


© 2015 Altova GmbH

2.5.2 OnUpdateCommand

Declaración: `OnUpdateCommand(nID as long, pXMLSpy as IDispatch) as SPYUpdateAction`

Descripción

Al método `OnUpdateCommand()` se le llama cada vez que se debe establecer el estado visible de un botón o de un comando de menú. `nID` almacena el ID del comando definido por el elemento `ID` del correspondiente elemento `UIElement`

`pXMLSpy` almacena una referencia a la interfaz de envío del objeto `Application`.

Los valores devueltos posibles para establecer el estado de actualización son:

```
spyEnable        = 1
spyDisable             = 2
spyCheck         = 4
spyUncheck       = 8
```

Ejemplo

```
Public Function IXMLSpyPlugIn_OnUpdateCommand(ByVal nID As Long, ByVal pXMLSpy
As Object) As SPYUpdateAction
    IXMLSpyPlugIn_OnUpdateCommand = spyDisable

    If (Not (pXMLSpy Is Nothing)) Then
        Dim objSpy As XMLSpyLib.Application
        Set objSpy = pXMLSpy

        If nID = 3 Or nID = 5 Then
            IXMLSpyPlugIn_OnUpdateCommand = spyEnable
        End If
        If nID = 4 Or nID = 6 Then
            If objSpy.Documents.Count > 0 Then
                IXMLSpyPlugIn_OnUpdateCommand = spyEnable
            Else
                IXMLSpyPlugIn_OnUpdateCommand = spyDisable
            End If
        End If
    End If
End Function
```

2.5.3 OnEvent

Declaración: `OnEvent(nEventID` as long, `arrayParameters` as `SAFEARRAY(VARIANT)`, `pXMLSpy` as `IDispatch)` as VARIANT

Descripción
Al método `OnEvent()` se le llama cada vez que se genera un evento desde Authentic Desktop.

Los valores posibles para `nEventID` son:

On_BeforeStartEditing	= 1
On_EditingFinished	= 2
On_FocusChanged	= 3
On_Beforedrag	= 4
On_BeforeDrop	= 5
On_OpenProject	= 6
On_OpenDocument	= 7
On_CloseDocument	= 8
On_SaveDocument	= 9

Eventos disponibles desde la versión 4r4 de Authentic Desktop:

On_DocEditDragOver	= 10
On_DocEditDrop	= 11
On_DocEditKeyDown	= 12
On_DocEditKeyUp	= 13
On_DocEditKeyPressed	= 14
On_DocEditMouseMove	= 15
On_DocEditButtonUp	= 16
On_DocEditButtonDown	= 17
On_DocEditContextMenu	= 18
On_DocEditPaste	= 19
On_DocEditCut	= 20
On_DocEditCopy	= 21
On_DocEditClear	= 22
On_DocEditSelectionChanged	= 23

Eventos disponibles desde la versión 2004 de Authentic Desktop:

On_DocEditDragOver	= 10

© 2015 Altova GmbH

Eventos disponibles desde la versión 2004r4 de Authentic Desktop (biblioteca de tipos versión 1.4):

On_BeforeOpenProject	= 25
On_BeforeOpenDocument	= 26
On_BeforeSaveDocument	= 27
On_BeforeCloseDocument	= 28
On_ViewActivation	= 29
On_DocEditKeyboardEvent	= 30
On_DocEditMouseEvent	= 31

Eventos disponibles desde la versión 2006 SP1 de Authentic Desktop (biblioteca de tipos versión 1.5):

On_BeforeValidate	= 32

Eventos disponibles desde la versión 2007 de Authentic Desktop (biblioteca de tipos versión 1.6):

On_BeforeShowSuggestions	= 33
On_ProjectOpened	= 34
On_Char	= 35

Eventos disponibles desde la versión 2009 de Authentic Desktop (biblioteca de tipos versión 2.2):

On_Initialize	= 36
On_Running	= 37
On_Shutdown	= 38

Eventos disponibles desde la versión 2012 de Authentic Desktop (biblioteca de tipos versión 2.8):

On_AuthenticBeforeSave	= 39
On_AuthenticContextMenuActivated	= 40
On_AuthenticLoad	= 41
On_AuthenticToolbarButtonClicked	= 42
On_AuthenticToolbarButtonExecuted	= 43
On_AuthenticUserAddedXMLNode	= 44

Los eventos se llaman igual que en el entorno de scripting de Authentic Desktop. Para los complementos para entornos IDe los nombres utilizados son irrelevantes. Los eventos se identifican con el valor `ID`.

`arrayParameters` es una matriz que se rellena con los parámetros del evento generado actualmente. El orden, el tipo y el significado de cada parámetro está disponible en el entorno de scripting de Authentic Desktop. El módulo Eventos de un proyecto de scripting contiene funciones predefinidas para todos los eventos de todas las versiones anteriores a la versión 4.4. Los parámetros que se pasan a las funciones predefinidas son idénticos a los elementos de matriz del parámetro `arrayParameters`.

Los eventos generados desde la vista Authentic de Authentic Desktop no pasan ningún parámetro directamente. En su lugar se usan objetos de *evento*. A los objetos de evento se puede acceder desde el objeto *Document* del documento activo.

pXMLSpy almacena una referencia a la interfaz de envío del objeto `Application` de Authentic Desktop.

Si el valor devuelto de `OnEvent()` está definido, entonces el complemento para entornos IDE no obtendrá este evento después. Tampoco un controlador de eventos dentro del entorno de scripting. Recuerde que todos los complementos para entornos IDE obtienen/procesan el evento antes que el entorno de scripting.

© 2015 Altova GmbH

2.5.4 GetUIModifications

Declaración: `GetUIModifications()` as String

Descripción

Al método `GetUIModifications()` se le llama durante la inicialización del complemento para obtener los datos de configuración XML que definen los cambios en la interfaz gráfica de Authentic Desktop. Al método se le llama cuando el complemento se carga por primera vez y cada vez que se inicia Authentic Desktop.

Consulte el apartado *Configuración XML* para ver cómo se modifica la interfaz gráfica.

Ejemplo

```
Public Function IXMLSpyPlugIn_GetUIModifications() As String
    ' GetUIModifications() obtiene el archivo XML con las modificaciones de la
interfaz gráfica
    ' del archivo config.xml de la carpeta del complemento
    Dim strPath As String
    strPath = App.Path

    If Len(strPath) > 0 Then
        Dim fso As New FileSystemObject
        Dim file As file

        Set file = fso.GetFile(strPath & "\config.xml")

        If (Not (file Is Nothing)) Then
            Dim stream As TextStream
            Set stream = file.OpenAsTextStream(ForReading)

            ' esto reemplaza el símbolo '**path**' de archivo XML
            ' con la ruta de acceso de instalación del complemento para obtener
el archivo de imagen
            Dim strMods As String
            strMods = stream.ReadAll
            strMods = Replace(strMods, "**path**", strPath)

            IXMLSpyPlugIn_GetUIModifications = strMods
        Else
            IXMLSpyPlugIn_GetUIModifications = ""
        End If
    End If
End Function
```

2.5.5 GetDescription

Declaración: `GetDescription()` as String

Descripción

`GetDescription()` sirve para definir la cadena *Description* para las entradas del complemento que aparecen en el cuadro de diálogo "Personalizar".

Ejemplo

```
Public Function IXMLSpyPlugIn_GetDescription() As String
    IXMLSpyPlugIn_GetDescription = "Complemento de muestra para XMLSpy;Este
complemento demuestra la implementación de un DLL Visual Basic sencillo como
complemento para XMLSpy."
End Function
```


© 2015 Altova GmbH

3 Application API

The COM-based API of Authentic Desktop (also called the Application API from now on) enables other applications to use the functionality of Authentic Desktop. As a result, it is possible to automate a wide range of tasks, from validating an XML file to modifying complex XML content (with the XMLData interface).

Authentic Desktop and its Application API follow the common specifications for automation servers set out by Microsoft. It is possible to access the methods and properties of the Application API from common development environments, such as those using C, C++, VisualBasic, and Delphi, and with scripting languages like JScript and VBScript.

Execution environments for the Application API

The Application API can be accessed from the following execution environments:

- External programs (described below and in the Overview part of this section)
- From within the built-in Scripting Editor of Authentic Desktop. For a description of the scripting environment, see the section, Scripting Editor.
- Authentic Desktop allows you to create and integrate your own plug-ins into the application using a special interface for plug-ins. A description of how to create plug-ins is given in the section IDE Plug-ins.
- Via an ActiveX Control, which is available if the integration package is installed. For more information, see the section ActiveX Integration.

External programs

In the Overview part of this section, we describe how the functionality of Authentic Desktop can be accessed and automated from external programs.

Using the Application API from outside Authentic Desktop requires an instance of Authentic Desktop to be started first. How this is done depends on the programming language used. See the section, Programming Languages, for information about individual languages.

Essentially, Authentic Desktop will be started via its COM registration. Then the `Application` object associated with the Authentic Desktop instance is returned. Depending on the COM settings, an object associated with an already running Authentic Desktop can be returned. Any programming language that supports creation and invocation of COM objects can be used. The most common of these are listed below.

- JScript and VBScript script files have a simple syntax and are designed to access COM objects. They can be run directly from a DOS command line or with a double click on Windows Explorer. They are best used for simple automation tasks.
- C# is a a full-fledged programming language that has a wide range of existing functionality. Access to COM objects can be automatically wrapped using C#..
- C++ provides direct control over COM access but requires relatively larger amounts of code than the other languages.
- Java: Altova products come with native Java classes that wrap the Application API and provide a full Java look-and-feel.
- Other programming languages that make useful alternatives are: Visual Basic for Applications, Perl, and Python.

Programming points

The following limitations must be considered in your client code:

- Be aware that if your client code crashes, instances of Authentic Desktop may still remain in the system.
- Don't hold references to objects in memory longer than you need them, especially those from the `XMLData` interface. If the user interacts between two calls of your client, then there is no guarantee that these references are still valid.
- Don't forget to disable dialogs if the user interface is not visible.
- See Error handling in JScript (and in C# and Java) for details of how to avoid annoying error messages.
- Free references explicitly if you are using C or C++.

This documentation

This documentation section about the Application API is broadly divided into two parts.

- The first part consists of an Overview, which describes the object model for the API and explains how the API is accessed via various programming languages.
- The second part is a reference section (Interfaces and Enumerations) that contains descriptions of the interface objects of the Application API.

© 2015 Altova GmbH

3.1 Overview

This overview of the Application API is organized as follows:

- The Object Model describes the relationships between the objects of the Application API.
- Programming Languages explains how the most commonly used programming languages (JScript, VBScript, C#, and Java) can be used to access the functionality of the Application API. Code listings from the example files supplied with your application package are used to describe basic mechanisms.

3.1.1 Object Model

The starting point for every application which uses the Application API is the `Application` object. This object contains general methods like import/export support and references to the open documents and any open project.

To create an instance of the `Application` object, call `CreateObject("Authentic Desktop.Application")` from VisualBasic or a similar function from your preferred development environment to create a COM object. There is no need to create any other objects in order to use the complete Application API; it is in fact not even possible. All other interfaces are accessed through other objects with the `Application` object as the starting point.

XMLSpy.Application or AuthenticDesktop.Application

Authentic Desktop installs a TypeLibrary containing the XMLSpyLib. If this TypeLibrary has been added to the development environment (VB development environment, for example) then an object of the `Application` type can be created with: `Set objSpy = New XMLSpyLib.Application`

If only Authentic Desktop is installed (and not XMLSpy), then

```
Set objSpy = GetObject("", "XMLSpy.Application")
```

does **not** work, because there won't be any object registered in the Registry with a ProgID of `XMLSpy.Application`. In this case, the registered object `is AuthenticDesktop.Application`.

The code listings in this documentation assume that both Authentic Desktop and XMLSpy have been installed. If, however, only Authentic Desktop has been installed, then please modify code fragments to take account of this difference.

The application object consists of the following parts:

1. Document collection and reference to the active document.
2. Reference to current project and methods for creating and opening projects.
3. Methods to support the export to and import from databases, text files, and Word documents.
4. URL management.
5. Methods for macro menu items.

Once you have created an `Application` object you can start using the functionality of Authentic Desktop. In most cases, you either open a project and access the documents from there or you directly open a document via the `Documents` interface.

© 2015 Altova GmbH

3.1.2 Programming Languages

Programming languages differ in the way they support COM access. A few examples for the most frequently used languages (*links below*) will help you get started. The code listings in this section show how basic functionality can be accessed. This basic functionality is included in the files in the API Examples folder and can be tested straight away. The path to the API Examples folder is given below:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

JScript

The JScript listings demonstrate the following basic functionality:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling
- Events

VBScript

VBScript is different than JScript only syntactically; otherwise it works in the same way. The listings below describe is an example of how VBScript can be used. For more information, refer to the JScript examples.

- Events: Shows how events are handled using VBScript.

C#

C# can be used to access the Application API functionality. The code listings show how to access the API for certain basic functionality.

- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates the application if it is already running.
- Open OrgChart.pxf: Locates one of the example documents installed with Authentic Desktop and opens it. If this document is already open it becomes the active document.
- OnDocumentOpened Event On/Off: Shows how to listen to Authentic Desktop events. When turned on, a message box will pop up after a document has been opened.
- Open ExpReport.xml: Opens another example document.
- Toggle View Mode: Changes the view of all open documents between Browser View and Authentic View. The code shows how to iterate through open documents.
- Validate: Validates the active document and shows the result in a message box. The code shows how to handle errors and COM output parameters.
- Shutdown Authentic Desktop: Stops Authentic Desktop.

Java

The Authentic Desktop API can be accessed from Java code. The Java sub-section of this section

explains how some basic Authentic Desktop functionality can be accessed from Java code. It is organized into the following sub-sections:

- Mapping Rules for the Java Wrapper
- Example Java Project
- Application Startup and Shutdown
- Simple Document Access
- Iterations
- Use of Out-Parameters
- Event Handlers

JScript

This section contains listings of JScript code that demonstrate the following basic functionality:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling
- Events
- Example: Bubble Sort Dynamic Tables

Example files

The code listings in this section are available in example files that you can test as is or modify to suit your needs. The JScript example files are located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

The example files can be run in one of two ways:

- *From the command line:* Open a command prompt window and type the name of one of the example scripts (for example, `Start.js`). The Windows Scripting Host that is packaged with newer versions of Windows (XP, Vista, 7) will execute the script.
- *From Windows Explorer:* In Windows Explorer, browse for the JScript file and double-click it. The Windows Scripting Host that is packaged with newer versions of Windows (XP, Vista, 7) will execute the script. After the script is executed, the command console gets closed automatically.

Start Application

The JScript below starts the application and shuts it down. If an instance of the application is already running, the running instance will be returned.

Script listing

The JScript listing below is explained with comments in the code.

© 2015 Altova GmbH

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object
of an already
// running application might be returned.
try
{
	objAuthentic = WScript.GetObject("", "AuthenticDesktop.Application");
}
catch(err)
{
	Exit("Can't access or create AuthenticDesktop.Application");
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objAuthentic.Visible = true;

WScript.Echo("Hello");

objAuthentic.Visible = false; // will shutdown application if it has no more COM
connections
//objAuthentic.Visible = true;	// will keep application running with UI
visible
```

Running the script

The JScript code listed above is available in the file `Start.js` located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

To run the script, start it from a command prompt window or from Windows Explorer.

Simple Document Access

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and close documents.

Running the script

The JScript code listed below is available in the file `DocumentAccess.js` located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7,	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

Windows 8	

To run the script, start it from a command prompt window or from Windows Explorer.

Script listing

The JScript listing below is explained with comments in the code.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object
of an already
// running application might be returned.
try
{
	objAuthentic = WScript.GetObject("", "AuthenticDesktop.Application");
}
catch(err)
{
	Exit("Can't access or create AuthenticDesktop.Application");
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objAuthentic.Visible = true;

// ***************************** code snippet for "Simple Document Access"
****************************************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted
to major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\Authentic2012\\AuthenticExamples\\";

// Tell Authentic to open two documents. No dialogs
objDoc1 = objAuthentic.Documents.OpenFile(strExampleFolder + "OrgChart.pxf",
false);
objAuthentic.Documents.OpenFile(strExampleFolder + "ExpReport.xml", false);

// The document currently active can be easily located.
objDoc2 = objAuthentic.ActiveDocument;

// Let us make sure that the document is shown in grid view.
objDoc2.SwitchViewMode(5);  // SPYViewModes.spyViewAuthentic = 5

// Now switch back to the document opened first
objDoc1.SetActiveDocument();

// ***************************** code snippet for "Simple Document Access"
****************************************

// ***************************** code snippet for "Iteration"
****************************************

// go through all open documents using a JScript Enumerator
```


© 2015 Altova GmbH

```
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objAuthentic.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
	if (iterDocs.item().IsModified)
		bRequiresSaving = true;

	var strErrorText = new Array(1);
	var nErrorNumber = new Array(1);
	var errorData = new Array(1);

	if (!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
	{
	var text = strErrorText;
	   // access that XMLData object only if filled in
	if (errorData[0] != null)
			text += "(" + errorData[0].Name + "/" +
errorData[0].TextValue + ")";

		WScript.Echo("Document \"" + iterDocs.item().Name +"\" validation
error[" + nErrorNumber + "]: " + text);
	}
	else
	{
		// The COM call succeeded and the document is valid.
		WScript.Echo("Document \"" + iterDocs.item().Name + "\" is
valid.");
	}
}

// go through all open documents using index-based access to the document
collection
for (i = objAuthentic.Documents.Count; i > 0; i--)
	objAuthentic.Documents.Item(i).Close(false);


// ************************** code snippet for "Iteration"
****************************************

//objAuthentic.Visible = false;		// will shutdown application if it has
no more COM connections
objAuthentic.Visible = true;	// will keep application running with UI
visible
```

Iteration

The JScript listing below shows how to iterate through the open documents.

Running the script

You can test this script by copying the listing below to a file, naming the file with a `.js` extension, and running the file from either the command line or Windows Explorer. You could copy the file to the `JScript` folder of the API Examples folder:

Windows XP	C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/
Windows Vista, Windows 7, Windows 8	C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/

To run the script, start it from a command prompt window or from Windows Explorer.

Script listing

The JScript listing below is explained with comments in the code.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM object
of an already
// running application might be returned.
try
{
	objAuthentic = WScript.GetObject("", "AuthenticDesktop.Application");
}
catch(err)
{
	Exit("Can't access or create AuthenticDesktop.Application");
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objAuthentic.Visible = true;

// ************************** code snippet for "Simple Document Access"
****************************************

// Locate examples via USERPROFILE shell variable. The path needs to be adapted
to major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\Authentic2012\\AuthenticExamples\\";

// Tell Authentic to open two documents. No dialogs
objDoc1 = objAuthentic.Documents.OpenFile(strExampleFolder + "OrgChart.pxf",
false);
objAuthentic.Documents.OpenFile(strExampleFolder + "ExpReport.xml", false);

// The document currently active can be easily located.
objDoc2 = objAuthentic.ActiveDocument;

// Let us make sure that the document is shown in grid view.
objDoc2.SwitchViewMode(1);  // SPYViewModes.spyViewText = 1

// Now switch back to the document opened first
objDoc1.SetActiveDocument();

// ************************** code snippet for "Simple Document Access"
****************************************
```


© 2015 Altova GmbH

```
// **************************** code snippet for "Iteration"
****************************************

// go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objAuthentic.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
	if (iterDocs.item().IsModified)
		bRequiresSaving = true;

	var strErrorText = new Array(1);
	var nErrorNumber = new Array(1);
	var errorData = new Array(1);

	if (!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
	{
	var text = strErrorText;
	   // access that XMLData object only if filled in
	if (errorData[0] != null)
			text += "(" + errorData[0].Name + "/" +
errorData[0].TextValue + ")";

		WScript.Echo("Document \"" + iterDocs.item().Name +"\" validation
error[" + nErrorNumber + "]: " + text);
	}
	else
	{
		// The COM call succeeded and the document is valid.
		WScript.Echo("Document \"" + iterDocs.item().Name + "\" is
valid.");
	}
}

// go through all open documents using index-based access to the document
collection
for (i = objAuthentic.Documents.Count; i > 0; i--)
	objAuthentic.Documents.Item(i).Close(false);


// **************************** code snippet for "Iteration"
****************************************

//objAuthentic.Visible = false;		// will shutdown application if it has
no more COM connections
objAuthentic.Visible = true;	// will keep application running with UI
visible
```

Error Handling

The Application API returns errors in two different ways:

- The `HRESULT` returned by every API method
- The `IErrorInfo` interface of the Application API

Every API method returns an HRESULT. This return value gives the caller information about errors during execution of the method. If the call was successful, the return value is S_OK. The HRESULT option is commonly used in C/C++ programs.

However, programming languages such as VisualBasic and scripting languages (and other high-level development environments) don't give the programmer access to the HRESULT return of a COM call. Such languages use the IErrorInfo interface, which is also supported by the Application API. If an error occurs, the Application API creates a new object that implements the IErrorInfo interface. The information provided by the IErrorInfo interface is imported by the development environment into its own error-handling mechanism.

JScript error handling

JScript provides a try-catch mechanism to deal with errors raised from COM calls. An error object containing the necessary information is declared. The listing below shows how.

```
// go through all open documents using a JScript Enumerator
bRequiresSaving = false;
for (var iterDocs = new Enumerator(objAuthentic.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
	if (iterDocs.item().IsModified)
		bRequiresSaving = true;

	var strErrorText = new Array(1);
	var nErrorNumber = new Array(1);
	var errorData = new Array(1);

	if (!iterDocs.item().IsValid(strErrorText, nErrorNumber, errorData))
	{
	var text = strErrorText;
	   // access that XMLData object only if filled in
	if (errorData[0] != null)
			text += "(" + errorData[0].Name + "/" +
errorData[0].TextValue + ")";

		WScript.Echo("Document \"" + iterDocs.item().Name +"\" validation
error[" + nErrorNumber + "]: " + text);
	}
	else
	{
		// The COM call succeeded and the document is valid.
		WScript.Echo("Document \"" + iterDocs.item().Name + "\" is
valid.");
	}
}
```

Events

COM specifies that a client must register itself at a server for callbacks using the connection point mechanism. The automation interface for XMLSpy defines the necessary event interfaces. The way to connect to those events depends on the programming language you use in your client. The following code listing shows how this is done using JScript.

© 2015 Altova GmbH

The method `WScript.ConnectObject` is used to receive events.

```
// The event-handler function
function DocEvent_OnBeforeCloseDocument(objDocument)
{
	WScript.Echo("Received event - before closing document");
}

// Create or connect to XMLSpy (or Authentic Desktop)
try
{
	// Create the environment and XMLSpy (or Authentic Desktop)
	objWshShell = WScript.CreateObject("WScript.Shell");
	objFSO = WScript.CreateObject("Scripting.FileSystemObject");
	objSpy = WScript.GetObject("", "XMLSpy.Application");

// If only Authentic Desktop is installed (and XMLSpy is not installed) use:
// objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

}
catch(err)
	{ WScript.Echo ("Can't create WScript.Shell object or XMLSpy"); }

// Create document object and connect to its events
objSpy.Visible = true;
objDoc = objSpy.Documents.OpenFile ("C:\\Program Files\\Altova\\Authentic2012\
\AuthenticExamples\\OrgChart.xml", false);
WScript.ConnectObject(objDoc, "DocEvent_");

// Keep running while waiting for the event
// In the meanwhile close this document in XMLSpy (or Authentic Desktop)
manually
WScript.Echo ("Sleeping for 10 seconds ...");
WScript.Sleep (10000);

objDoc = null;
WScript.Echo ("Stopped listening for event");
objSpy.Quit();
```

Example: Bubble Sort Dynamic Tables

The following JScript snippet will sort any dynamic table by the table column identified by the current cursor position. The sort process is performed on screen. The undo buffer is available for all performed operations.

If you can run JScript on you computer - as you most likely will - copy the following code into a file with extension 'js'. Execute the script by double-clicking it in Windows Explorer, or run it from the command line.

```
// some useful XMLSpy enum constants
var spyAuthenticTag = 6;
var spyAuthenticTable = 9;
var spyAuthenticTableRow = 10;
var spyAuthenticTableColumn = 11;
```

```
var spyAuthenticRangeBegin = 2;

// example call for the sort table function
try
{
  var objSpy = WScript.GetObject("", "XMLSpy.Application");
  // If only Authentic is installed (and XMLSpy is not installed) use:
  // var objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

  // use current selection to indicate which column to sort
  SortCurrentTable (objSpy.ActiveDocument.AuthenticView.Selection);
}
catch (err)
  { WScript.Echo ("Please open a document in authentic view, and select a table
  column\n" +
                  "Error : (" + (err.number & 0xffff) + ")" + err.description);
}

// we assume that XMLSpy is running, a document with a dynamic table
// is open, and the cursor is in a table column that will
// be used for sorting.
function SortCurrentTable (objCursor)
{
  if (objCursor.IsInDynamicTable())
  {
    // calculate current column index
    var nColIndex = 0;
    while (true)
    {
      // go left column-by-column
      try { objCursor.GotoPrevious(spyAuthenticTableColumn); }
      catch (err) { break; }
      nColIndex++;
    }

    // count number of table rows, so the bubble loops become simpler.
    // goto begin of table
    var objTableStart =
    objCursor.ExpandTo(spyAuthenticTable).CollapsToBegin().Clone();
    var nRows = 1;
    while (true)
    {
      // go down row-by-row
      try { objTableStart.GotoNext(spyAuthenticTableRow); }
      catch (err) { break; }
      nRows++;
    }

    // bubble sort through table
    for (var i = 0; i < nRows - 1; i++)
    {
      // select correct column in first table row
      var objBubble =
      objCursor.ExpandTo(spyAuthenticTable).CollapsToBegin().Clone();
```


© 2015 Altova GmbH

```
      objBubble.Goto (spyAuthenticTableColumn, nColIndex,
      spyAuthenticRangeBegin).ExpandTo(spyAuthenticTag);

      // bubble this row down as far as necessary
      for (var j = 0; j < nRows - i - 1; j++)
      {
        var strField1 = objBubble.Text;
        // now look for the comparison table cell: start of next row and right of
        the correct column
        var strField2 = objBubble.GotoNext(spyAuthenticTableRow).
                                  Goto (spyAuthenticTableColumn, nColIndex,
        spyAuthenticRangeBegin).
                                  ExpandTo(spyAuthenticTag).Text;
        if (strField1 > strField2)
        {
          objBubble.MoveRowUp();  // swap the rows
          // and re-calculate objBubble to select the cell to bubble
          objBubble.GotoNext(spyAuthenticTableRow).
                    Goto (spyAuthenticTableColumn, nColIndex,
          spyAuthenticRangeBegin).
                    ExpandTo(spyAuthenticTag);
        }
      }
    }
  }
  else
    WScript.Echo ("please, select a table cell first");
}
```

VBScript

VBScript is syntactically different than JScript but works in the same way. This section contains a listing showing how events are used with VBScript and an example.

For information about other functionality, refer to the JScript examples listed below:

- Start application or attach to a running instance
- Simple document access
- Iteration
- Error handling

Events

COM specifies that a client must register itself at a server for callbacks using the connection point mechanism. The automation interface for XMLSpy defines the necessary event interfaces. The way to connect to those events depends on the programming language you use in your client. The following code listing shows how this is done using VBScript.

The method `WScript.ConnectObject` is used to receive events.

```
' the event handler function
Function DocEvent_OnBeforeCloseDocument(objDocument)
   Call WScript.Echo("received event - before closing document")
End Function
```

```
    ' create or connect to XMLSPY
    Set objWshShell = WScript.CreateObject("WScript.Shell")
    Set objFSO = WScript.CreateObject("Scripting.FileSystemObject")
    Set objSpy = WScript.GetObject("", "XMLSpy.Application")
    ' If only Authentic is installed (and XMLSpy is not installed) use:
    ' Set objSpy = WScript.GetObject("", "AuthenticDesktop.Application")

    ' create document object and connect to its events
    objSpy.Visible = True
    Set objDoc = objSpy.Documents.OpenFile ("C:\\Program Files\\Altova\
    \XMLSPY2004\\Examples\\OrgChart.xml", False)
    Call WScript.ConnectObject(objDoc, "DocEvent_")

    ' keep running while waiting on the event
    ' in the meantime close the document in XMLSPY manually
    Call WScript.Echo ("sleeping for 10 seconds ...")
    Call WScript.Sleep (10000)

    Set objDoc = Nothing
    Call WScript.Echo ("stopped listening for event")
    Call objSpy.Quit
```

Example: Using Events

Authentic View supports event connection on a per-object basis. Implementation of this feature is based on COM connection points and is available in environments that support this mechanism.

The following example is a VBScript code example that shows how to use events from within a VBScript project.

```
Rem ------------------------------------------------------------------------
Rem VBScript example that demonstrates how to use events.
Rem ------------------------------------------------------------------------

' Event handler for OnSelectionChanged event of AuthenticView
Function AuthenticViewEvent_OnSelectionChanged(objAuthenticRange)
    If objAuthenticRange.FirstTextPosition <>
objAuthenticRange.LastTextPosition Then
        Call WScript.Echo("Selection: " & objAuthenticRange.Text & vbNewLine &
vbNewLine & "Close this dialog.")
    Else
        Call WScript.Echo("Cursor position: " &
objAuthenticRange.FirstTextPosition & vbNewLine & vbNewLine & "Close this
dialog.")
    End If
End Function

' Here starts the main code.
' Find out user's personal folder and locate one of the installed XMLSpy 2011
examples.
Set WshShell = WScript.CreateObject("WScript.Shell")
personalFolder = WshShell.ExpandEnvironmentStrings("%UserProfile%")
xmlspyExamplesFolder = personalFolder & "\My Documents\Altova\XMLSpy2011
\Examples\"
docPath = xmlspyExamplesFolder & "OrgChart.xml"
```


© 2015 Altova GmbH

```
' Start/access XMLSpy and connect to its automation interface.
Set objSpy = GetObject("", "XMLSpy.Application")
' Make the UI of XMLSpy visible.
objSpy.Visible = True

' Create object to access windows file system and test if the our document
exists.
Set fso = CreateObject("Scripting.FileSystemObject")
If fso.FileExists(docPath) Then
    ' open the document
    Call objSpy.Documents.OpenFile(docPath, False)
    set objDoc = objSpy.ActiveDocument

    ' switch active document to authentic view
    objDoc.SwitchViewMode 4 ' spyViewAuthentic

    ' Register for connection point events on the authentic view of the active
document.
    ' Any function with a valid event name prefixed with "AuthenticViewEvent_"
will
    ' be called when the corresponding event gets triggered on the specified
object.
    set objView = objDoc.AuthenticView
    Call WScript.ConnectObject(objView, "AuthenticViewEvent_")
    Call WScript.Echo("Events are connected." & vbNewLine & vbNewLine & "Now set
or move the cursor in XMLSpy." & vbNewLine & vbNewLine & "Close this dialog to
shut down XMLSpy.")

    ' To disconnect from the events delete the reference to the object.
    set objView = Nothing
Else
    Call WScript.Echo("The file " & docPath & " does not exist.")
End If

' shut down XMLSpy when this script ends
objSpy.Visible = False
```

C#

The C# programming language can be used to access the Application API functionality. You could use Visual Studio 2008 or Visual Studio 2010 to create the C# code, saving it in a Visual Studio project. Create the project as follows:

1. In Microsoft Visual Studio, add a new project using **File | New | Project**.
2. Add a reference to the Authentic Desktop Type Library by clicking **Project | Add Reference**. The Add Reference dialog appears. Browse for the Authentic Desktop Type Library component, which is located in the Authentic Desktop application folder, and add it.
3. Enter the code you want.
4. Compile the code and run it.

Example C# project

Your Authentic Desktop package contains an example C# project, which is located in the C# folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

The code listing below shows how basic application functionality can be used. This code is similar to the example C# project in the API Examples folder of your application package, but might differ slightly.

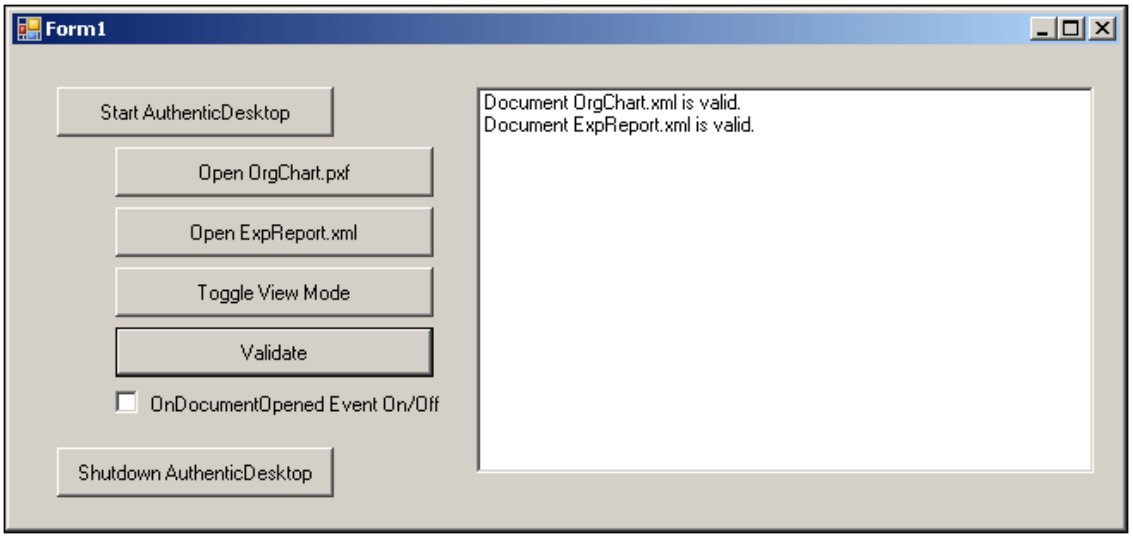
Platform configuration

If you have a 64-bit operating system and are using a 32-bit installation of Authentic Desktop, you must add the x86 platform in the solution's Configuration Manager and build the sample using this configuration.

A new x86 platform (for the active solution in Visual Studio) can be created in the New Solution Platform dialog (**Build | Configuration Manager | Active solution platform | <New...>**).

What the code listing below does

The example code listing below creates a simple user interface (*screenshot below*) with buttons that invoke basic Authentic Desktop operations:



- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates the application if it is already running.
- Open OrgChart.pxf: Locates one of the example documents installed with Authentic Desktop and opens it. If this document is already open it becomes the active document.
- Open ExpReport.xml: Opens another example document.
- Toggle View Mode: Changes the view of all open documents between Text View and Authentic View. The code shows how to iterate through open documents.

© 2015 Altova GmbH

- Validate: Validates the active document and shows the result in a message box. The code shows how to handle errors and COM output parameters.
- Shut down Authentic Desktop: Stops Authentic Desktop.

You can modify the code (of the code listing below or of the example C# project in the API Examples folder) in any way you like and run it.

Compiling and running the example

In the API Examples folder, double-click the file `AutomateAuthenticDesktop_VS2008.sln` (to open it in Visual Studio 2008) or the file `AutomateAuthenticDesktop_VS2010.sln` (to open it in Visual Studio 2010). Alternatively the file can be opened from within Visual Studio (with **File | Open | Project/Solution**). To compile and run the example, select **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Code listing of the example

Given below is the C# code listing of the basic functionality of the form (`Form1.cs`) created in the `AutomateAuthenticDesktop` example. Note that the code listed below might differ slightly from the code in the API Examples form.The listing below is commented for ease of understanding. Parts of the code are also presented separately in the sub-sections of this section, according to the Application API functionality they access.

The code essentially consists of a series of handlers for the buttons in the user interface shown in the screenshot above.

```
namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of AuthenticDesktop is accessed via its automation
interface.
        XMLSpyLib.Application AuthenticDesktop;

        // Location of examples installed with AuthenticDesktop
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
            // Locate examples installed with AuthenticDesktop.
            // REMARK: You might need to adapt this if you have a different
major version of the product.
            strExamplesFolder =
Environment.GetEnvironmentVariable("USERPROFILE") + "\\My Documents\\Altova\
\Authentic2012\\AuthenticExamples\\";
        }

        // Handler for the "Start AuthenticDesktop" button
```

```
        private void StartAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // If there is no AuthenticDesktop instance, create one and make
it visible.
                AuthenticDesktop = new XMLSpyLib.Application();
                AuthenticDesktop.Visible = true;

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // If an AuthenticDesktop instance is already running, make sure
it's visible.
                if (!AuthenticDesktop.Visible)
                    AuthenticDesktop.Visible = true;
            }
        }

        // Handler for the "Open OrgChart.pxf" button
        private void openOrgChart_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that
it's visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder +
"OrgChart.pxf", false);
        }

        // Handler for the "Open ExpReport.xml" button
        private void openExpReport_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that
it's visible
            StartAuthenticDesktop_Click(null, null);

            // Open a sample file installed with the product.
            AuthenticDesktop.Documents.OpenFile(strExamplesFolder +
"ExpReport.xml", false);
        }

        // Handler for the "Toggle View Mode" button
        private void toggleView_Click(object sender, EventArgs e)
        {
            // Make sure there's a running Authentic Desktop instance, and that
it's visible
            StartAuthenticDesktop_Click(null, null);

            // Iterate through all open documents and toggle the current view
between Text View and Authentic View.
            foreach (XMLSpyLib.Document doc in AuthenticDesktop.Documents)
                if (doc.CurrentViewMode ==
XMLSpyLib.SPYViewModes.spyViewAuthentic)
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewBrowser);
                else
```

 © 2015 Altova GmbH

```
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
        }

        // Handler for the "Shut down AuthenticDesktop" button
        // Shut down application instance by explicitely releasing the COM
object.
        private void shutdownAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                // Allow shut down of AuthenticDesktop by releasing the UI
                AuthenticDesktop.Visible = false;

                // Explicitly release the COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(AuthenticDesktop) >
0) ;
                }
                finally
                {
                    // Avoid subsequent access to this object.
                    AuthenticDesktop = null;
                }
            }
        }

        // Handler for the "Validate" button
        private void validate_Click(object sender, EventArgs e)
        {
            // COM errors get returned to C# as exceptions. Use a try/catch
block to handle them.
            try
            {
                // Method 'IsValid' is one of the few functions that use output
parameters.
                // Use 'object' type for these parameters.
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText,
ref nErrorNumber, ref errorData))
                {
                    // The COM call succeeds but the document is not valid.
                    // A detailed description of the problem is returned in
strErrorText, nErrorNumber and errorData.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData !=
null ? ((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeds and the document is valid.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is valid.");
```

```
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no
document is open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }

        delegate void addListBoxItem_delegate(string sText);
        // Called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // Wrapper method to call UI control methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate
methodToInvoke, String sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread
waits for completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling
thread doesn't wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "An empty document was created.";

            // Synchronize the calling thread with the UI thread because
            // COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // Call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from
COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method
which will be called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }


        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs
e)
        {
            if (AuthenticDesktop != null)
            {
                if (checkBoxEventOnOff.Checked)
```


© 2015 Altova GmbH

```
                    AuthenticDesktop.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpene
d);
                else
                    AuthenticDesktop.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpene
d);
            }
        }
    }
}
```

Add Reference to Authentic Desktop API

Add the application's type library as a reference in a .NET project as follows: With the .NET project open, click **Project | Add Reference**. Then browse for the type library, which is called `Authentic Desktop.tlb`, and is located in the Authentic Desktop application folder.

Then declare a variable to access the Authentic Desktop API:

```
        // An instance of Authentic Desktop is accessed via its automation
interface.
        XMLSpyLib.Application Authentic Desktop;
```

Application Startup and Shutdown

In the code snippets below, the methods `StartAuthenticDesktop_Click` and `ShutdownAuthenticDesktop_Click` are those assigned to buttons in the AutomateAuthenticDesktop example that, respectively, start up and shut down the application. This example is located in the C# folder of the API Examples folder (*see the file Form1.cs*):

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

Starting Authentic Desktop

The following code snippet from the AutomateAuthenticDesktop example shows how to start up the application.

```
        // Handler for the "Start AuthenticDesktop" button
        private void StartAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // If there is no AuthenticDesktop instance, create one and make
```

```
it visible.
                AuthenticDesktop = new XMLSpyLib.Application();
                AuthenticDesktop.Visible = true;

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // If an instance of Authentic Desktop is already running, make
sure it's visible
                if (!AuthenticDesktop.Visible)
                    AuthenticDesktop.Visible = true;
            }
        }
```

Shutting down Authentic Desktop

The following code snippet from the AutomateAuthenticDesktop example shows how to shut down the application.

```
        // Handler for the "Shut down AuthenticDesktop" button
        // Shut down application instance by explicitely releasing the COM
object.
        private void shutdownAuthenticDesktop_Click(object sender, EventArgs e)
        {
            if (AuthenticDesktop != null)
            {
                // Allow shut down of AuthenticDesktop by releasing the UI
                AuthenticDesktop.Visible = false;

                // Explicitly release the COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(AuthenticDesktop) >
0) ;
                }
                finally
                {
                    // Avoid subsequent access to this object.
                    AuthenticDesktop = null;
                }
            }
        }
```

Opening Documents

The code snippets below (from the AutomateAuthenticDesktop example) show how two files are opened via two separate methods assigned to two buttons in the user interface. Both methods use the same Application API access mechanism: `Documents.OpenFile(string, boolean)`.

The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/`

© 2015 Altova GmbH

	Altova/Authentic2016/AuthenticExamples/API/
Windows Vista, Windows 7, Windows 8	C:/Users/<username>/Documents/ Altova/Authentic2016/AuthenticExamples/API/

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

Code snippet

```
        // Handler for the "Open OrgChart.pxf" button
         private void openOrgChart_Click(object sender, EventArgs e)
         {
             // Make sure there's a running Authentic Desktop instance, and that
it's visible
             StartAuthenticDesktop_Click(null, null);

             // Open a sample file installed with the product.
             AuthenticDesktop.Documents.OpenFile(strExamplesFolder +
"OrgChart.pxf", false);
         }

         // Handler for the "Open ExpReport.xml" button
         private void openExpReport_Click(object sender, EventArgs e)
         {
             // Make sure there's a running Authentic Desktop instance, and that
it's visible
             StartAuthenticDesktop_Click(null, null);

             // Open a sample file installed with the product.
             AuthenticDesktop.Documents.OpenFile(strExamplesFolder +
"ExpReport.xml", false);
         }
```

The file opened last will be the active file.

Iterating through Open Documents

The code snippet below (from the AutomateAuthenticDesktop example; *see the file Form1.cs*) shows how to iterate through open documents. A condition is then tested within the iteration loop, and the document view is switched between Text View and Authentic View.

```
        // Handler for the "Toggle View Mode" button
         private void toggleView_Click(object sender, EventArgs e)
         {
             // Make sure there's a running Authentic Desktop instance, and that
it's visible
             StartAuthenticDesktop_Click(null, null);

             // Iterate through all open documents and toggle the current view
between Text View and Authentic View.
             foreach (XMLSpyLib.Document doc in AuthenticDesktop.Documents)
                 if (doc.CurrentViewMode ==
XMLSpyLib.SPYViewModes.spyViewAuthentic)
                     doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewBrowser);
```

```
                else
                    doc.SwitchViewMode(XMLSpyLib.SPYViewModes.spyViewAuthentic);
        }
```

The AutomateAuthenticDesktop example example is located in the C# folder of the API Examples folder:

Windows XP	C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/
Windows Vista, Windows 7, Windows 8	C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

Errors and COM Output Parameters

The code snippet below (from the AutomateAuthenticDesktop example) shows how to handle errors and COM output parameters. The method AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText, ref nErrorNumber, ref errorData) uses output parameters that are used, in the code snippet below, to generate an error-message text.

The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows XP	C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/
Windows Vista, Windows 7, Windows 8	C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

Code snippet

```
        // Handler for the "Validate" button
        private void validate_Click(object sender, EventArgs e)
        {
            // COM errors get returned to C# as exceptions. Use a try/catch
block to handle them.
            try
            {
                // Method 'IsValid' is one of the few functions that use output
parameters.
                // Use 'object' type for these parameters.
                object strErrorText = "";
                object nErrorNumber = 0;
                object errorData = null;

                if (!AuthenticDesktop.ActiveDocument.IsValid(ref strErrorText,
```

 © 2015 Altova GmbH

```
ref nErrorNumber, ref errorData))
                {
                    // The COM call succeeds but the document is not valid.
                    // A detailed description of the problem is returned in
strErrorText, nErrorNumber and errorData.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is not valid.");
                    listBoxMessages.Items.Add("\tErrorText  : " + strErrorText);
                    listBoxMessages.Items.Add("\tErrorNumber: " + nErrorNumber);
                    listBoxMessages.Items.Add("\tElement    : " + (errorData !=
null ? ((XMLSpyLib.XMLData)errorData).TextValue : "null"));
                }
                else
                {
                    // The COM call succeeds and the document is valid.
                    listBoxMessages.Items.Add("Document " +
AuthenticDesktop.ActiveDocument.Name + " is valid.");
                }
            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no
document is open.
                listBoxMessages.Items.Add("Error validating active document: " +
ex.Message);
            }
        }
```

Events

The code snippet below (from the AutomateAuthenticDesktop example) lists the code for two event handlers. The AutomateAuthenticDesktop example (*see the file Form1.cs*) is located in the C# folder of the API Examples folder:

Windows XP	C:/Documents and Settings/<username>/My Documents/ Altova/Authentic2016/AuthenticExamples/API/
Windows Vista, Windows 7, Windows 8	C:/Users/<username>/Documents/ Altova/Authentic2016/AuthenticExamples/API/

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

```
        delegate void addListBoxItem_delegate(string sText);
        // Called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // Wrapper method to call UI control methods from a worker thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate
methodToInvoke, String sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread
waits for completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling
```

```
thread doesn't wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // Event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(XMLSpyLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "An empty document was created.";

            // Synchronize the calling thread with the UI thread because
            // COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // Call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from
COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method
which will be called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }


        private void checkBoxEventOnOff_CheckedChanged(object sender, EventArgs
e)
        {
            if (AuthenticDesktop != null)
            {
                if (checkBoxEventOnOff.Checked)
                    AuthenticDesktop.OnDocumentOpened += new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpene
d);
                else
                    AuthenticDesktop.OnDocumentOpened -= new
XMLSpyLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpene
d);
            }
        }
```

Java

The Application API can be accessed from Java code. To allow accessing the Authentic Desktop automation server directly from Java code, the libraries listed below must reside in the `classpath`. They are installed in the folder: `JavaAPI` in the Authentic Desktop application folder.

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (`AltovaAutomation_x64.dll` in the case of 64-bit versions)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `AuthenticAPI.jar`: Java classes that wrap the Authentic Desktop automation interface
- `AuthenticAPI_JavaDoc.zip`: a `Javadoc` file containing help documentation for the Java API

© 2015 Altova GmbH

Note: In order to use the Java API, the DLL and Jar files must be on the Java Classpath.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details of the example Java project, see the section, Example Java Project.

Rules for mapping the Application API names to Java

The rules for mapping between the Application API and the Java wrapper are as follows:

- ***Classes and class names***
  For every interface of the Authentic Desktop automation interface a Java class exists with the name of the interface.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with `get` and `set` can be used. If a property does not support write-access, no setter method is available. Example: For the `Name` property of the `Document` interface, the Java methods `getName` and `setName` are available.

- ***Enumerations***
  For every enumeration defined in the automation interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus '`Event`' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `Application`: Java class to access the application
  `ApplicationEvents`: Events interface for the Application
  `ApplicationEventsDefaultHandler`: Default handler for `ApplicationEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Java name
`Document`, method `SetEncoding`	`setFileEncoding`
`AuthenticView`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Goto`	`gotoElement`
`AuthenticRange`, method `Clone`	`cloneRange`

This section

This section explains how some basic Authentic Desktop functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project
- Application Startup and Shutdown
- Simple Document Access
- Iterations
- Use of Out-Parameters
- Event Handlers

Example Java Project

The Authentic Desktop installation package contains an example Java project, located in the Java folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/Altova/Authentic2016/AuthenticExamples/API/`
Windows Vista, Windows 7, Windows 8	`C:/Users/<username>/Documents/Altova/Authentic2016/AuthenticExamples/API/`

This folder contains Java examples for the Authentic Desktop API. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below. If you are using a 64-bit version of the application, some filenames contain `_x64` in the name. These filenames are indicated with `(_x64)`.

`AltovaAutomation(_x64).dll`	Java-COM bridge: DLL part
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`AuthenticAPI.jar`	Java classes of the Authentic Desktop API
`RunAuthenticDesktop.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`Authentic_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

© 2015 Altova GmbH

What the example does

The example starts up Authentic Desktop and performs a few operations, including opening and closing documents. When done, Authentic Desktop stays open. You must close it manually.

- Start Authentic Desktop: Starts Authentic Desktop, which is registered as an automation server, or activates Authentic Desktop if it is already running.
- Open example files: Locates example documents installed with Authentic Desktop and opens them.
- Iteration and Changing the View Mode: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Iteration, validation, output parameters: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- Event Handling: Shows how to handle Authentic Desktop events.
- Shut down Authentic Desktop: Shuts down Authentic Desktop.

You can modify the example in any way you like and run it.

Running the example from the command line

To run the example from the command line, open a command prompt window, go to the Java folder of the API Examples folder (*see above for location*), and then type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a JDK 1.5 or later installation on your computer.

Press the **Return** key. The Java source in `RunAuthenticDesktop.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **Import | Existing Projects into Workspace** command to add the Eclipse project file (`.project`) located in the Java folder of the API Examples folder (*see above for location*). The project `RunAuthenticDesktop` will then appear in your Package Explorer or Navigator.

Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `RunAuthenticDesktop.java` is listed below with comments.

```
01 // Access general JAVA-COM bridge classes
02 import com.altova.automation.libs.*;
03
```

```
04 // Access AuthenticDesktop Java-COM bridge
05 import com.altova.automation.AuthenticDesktop.*;
06 import com.altova.automation.AuthenticDesktop.Enums.SPYViewModes;
07
08 /**
09  * A simple example that starts AuthenticDesktop COM server and performs a
view operations on it.
10  * Feel free to extend.
11  */
12 public class RunAuthenticDesktop
13 {
14   public static void main(String[] args)
15   {
16     // An instance of the application.
17     Application authenticDesktop = null;
18
19     // Instead of COM error-handling, use Java exception mechanism.
20     try
21     {
22       // Start AuthenticDesktop as COM server.
23       authenticDesktop = new Application();
24       // COM servers start up invisible so we make it visible
25       authenticDesktop.setVisible(true);
26
27       // Locate samples installed with the product.
28       String strExamplesFolder = System.getenv("USERPROFILE") + "\\My
Documents\\Altova\\Authentic2012\\AuthenticExamples\\";
29
30       // Open two files from the product samples.
31       authenticDesktop.getDocuments().openFile(strExamplesFolder +
"OrgChart.pxf", false);
32       authenticDesktop.getDocuments().openFile(strExamplesFolder +
"ExpReport.xml", false);
33
34       // Iterate through all open documents and set the View Mode to 'Text'.
35       for (Document doc:authenticDesktop.getDocuments())
36         if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
37           doc.switchViewMode(SPYViewModes.spyViewText);
38
39       // An alternative iteration mode is index-based. COM indices are
typically zero-based.
40       Documents documents = authenticDesktop.getDocuments();
41       for (int i = 1; i <= documents.getCount(); i++)
42       {
43         Document doc = documents.getItem(i);
44
45         // Validation is one of the few methods that have output parameters.
46         // The class JVariant is the correct type for parameters in these
cases.
47         // To get values back mark them with the by-reference flag.
48         JVariant validationErrorText = new JVariant.JStringVariant("");
validationErrorText.setByRefFlag();
49         JVariant validationErrorCount = new JVariant.JIntVariant(0);
validationErrorCount.setByRefFlag();
50         JVariant validationErrorXMLData = new JVariant.JIDispatchVariant(0);
 validationErrorXMLData.setByRefFlag();
51         if (!doc.isValid(validationErrorText, validationErrorCount,
validationErrorXMLData))
52           System.out.println("Document " + doc.getName() + " is not
```

 © 2015 Altova GmbH

```
wellformed - " + validationErrorText.getStringValue());
53          else
54            System.out.println("Document " + doc.getName() + " is
wellformed.");
55        }
56
57        // The following lines attach to the document events using a default
implementation
58        // for the events and override one of its methods.
59        // If you want to override all document events it is better to derive
your listener class
60        // from DocumentEvents and implement all methods of this interface.
61        Document doc = authenticDesktop.getActiveDocument();
62        doc.addListener(new DocumentEventsDefaultHandler()
63        {
64          @Override
65          public boolean onBeforeCloseDocument(Document i_ipDoc) throws
AutomationException
66          {
67            System.out.println("Document " + i_ipDoc.getName() + " requested
closing.");
68
69            // Allow closing of document
70            return true;
71          }
72        });
73        doc.close(true);
74        doc = null;
75
76        System.out.println("Watch AuthenticDesktop!");
77      }
78      catch (AutomationException e)
79      {
80        // e.printStackTrace();
81      }
82      finally
83      {
84        // Make sure that AuthenticDesktop can shut down properly.
85        if (authenticDesktop != null)
86          authenticDesktop.dispose();
87
88        // Since the COM server was made visible and still is visible, it will
keep running
89        // and needs to be closed manually.
90        System.out.println("Now close AuthenticDesktop!");
91      }
92    }
93 }
```

Application Startup and Shutdown

The code listings below show how the application can be started up and shut down.

Application startup

Before starting up the application, the appropriate classes must be imported (*see below*).

```
01 // Access general JAVA-COM bridge classes
02 import com.altova.automation.libs.*;
03
04 // Access AuthenticDesktop Java-COM bridge
05 import com.altova.automation.AuthenticDesktop.*;
06 import com.altova.automation.AuthenticDesktop.Enums.SPYViewModes;
07
08 /**
09  * A simple example that starts AuthenticDesktop COM server and performs a
view operations on it.
10  * Feel free to extend.
11  */
12 public class RunAuthenticDesktop
13 {
14   public static void main(String[] args)
15   {
16     // An instance of the application.
17     Application authenticDesktop = null;
18
19     // Instead of COM error-handling, use Java exception mechanism.
20     try
21     {
22       // Start AuthenticDesktop as COM server.
23       authenticDesktop = new Application();
24       // COM servers start up invisible so we make it visible
25       authenticDesktop.setVisible(true);
26
27 ...
28     }
29   }
30 }
```

Application shutdown

The application can be shut down as shown below.

```
1 {
2       // Make sure that AuthenticDesktop can shut down properly.
3       if (authenticDesktop != null)
4         authenticDesktop.dispose();
5
6       // Since the COM server was made visible and still is visible, it will
keep running
7       // and needs to be closed manually.
8       System.out.println("Now close AuthenticDesktop!");
9 }
```

Simple Document Access

The code listing below shows how to open a document.

```
1  // Locate samples installed with the product.
2  String strExamplesFolder = System.getenv("USERPROFILE") + "\\My Documents\
\Altova\\Authentic2012\\AuthenticExamples\\";
3
```


© 2015 Altova GmbH

```
4  // Open two files from the product samples.
5  authenticDesktop.getDocuments().openFile(strExamplesFolder + "OrgChart.pxf",
false);
6  authenticDesktop.getDocuments().openFile(strExamplesFolder + "ExpReport.xml",
false);
```

Iterations

The listing below shows how to iterate through open documents.

```
01 // Iterate through all open documents and set the View mode to 'Text'.
02 for (Document doc:authenticDesktop.getDocuments())
03   if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
04           doc.switchViewMode(SPYViewModes.spyViewText);
05
06 // An alternative iteration mode is index-based. COM indices are typically
zero-based.
07 Documents documents = authenticDesktop.getDocuments();
08 for (int i = 1; i <= documents.getCount(); i++)
09   {
10     Document doc = documents.getItem(i);
11     ...
12   }
```

Use of Out-Parameters

The code listing below iterates through open documents and validates each of them. For each validation, a message is generated using the output parameters of the Validation method.

```
01 // Iterate through all open documents and set the View mode to 'Text'.
02 for (Document doc:authenticDesktop.getDocuments())
03   if ( doc.getCurrentViewMode() != SPYViewModes.spyViewText)
04        doc.switchViewMode(SPYViewModes.spyViewText);
05
06 // An alternative iteration mode is index-based. COM indices are typically
zero-based.
07 Documents documents = authenticDesktop.getDocuments();
08 for (int i = 1; i <= documents.getCount(); i++)
09 {
10   Document doc = documents.getItem(i);
11
12 // Validation is one of the few methods that have output parameters.
13 // The class JVariant is the correct type for parameters in these cases.
14 // To get values back, mark them with the by-reference flag.
15 JVariant validationErrorText = new JVariant.JStringVariant("");
validationErrorText.setByRefFlag();
16 JVariant validationErrorCount = new JVariant.JIntVariant(0);
validationErrorCount.setByRefFlag();
17 JVariant validationErrorXMLData = new JVariant.JIDispatchVariant(0);
validationErrorXMLData.setByRefFlag();
18   if (!doc.isValid(validationErrorText, validationErrorCount,
validationErrorXMLData))
19     System.out.println("Document " + doc.getName() + " is not wellformed - "
+ validationErrorText.getStringValue());
```

```
20    else
21      System.out.println("Document " + doc.getName() + " is wellformed.");
22  }
```

Event Handlers

The listing below shows how to listen for and use events.

```
01 // The following lines attach to the document events using a default
implementation
02 // for the events and override one of its methods.
03 // If you want to override all document events, it is better to derive your
listener class
04 // from DocumentEvents and implement all methods of this interface.
05 Document doc = authenticDesktop.getActiveDocument();
06 doc.addListener(new DocumentEventsDefaultHandler()
07 {
08   @Override
09   public boolean onBeforeCloseDocument(Document i_ipDoc) throws
AutomationException
10   {
11     System.out.println("Document " + i_ipDoc.getName() + " requested
closing.");
12
13     // allow closing of document
14     return true;
15   }
16 });
17 doc.close(true);
18 doc = null;
```


© 2015 Altova GmbH

3.2 Interfaces

Object Hierarchy

Application
- SpyProject
  - SpyProjectItems
    - SpyProjectItem
- Documents
  - Document
    - GridView
    - AuthenticView
      - AuthenticRange
      - AuthenticDataTransfer (previously DocEditDataTransfer)
      - AuthenticDataTransfer (previously DocEditDataTransfer)
    - TextView
    - XMLData
- Dialogs
  - CodeGeneratorDlg
  - FileSelectionDlg
  - SchemaDocumentationDlg
  - GenerateSampleXMLDlg
  - DTDSchemaGeneratorDlg
  - FindInFilesDlg
- DatabaseConnection
- ExportSettings
- TextImportExportSettings
- ElementList
  - ElementListItem

Enumerations

Description

This chapter contains the reference of the Authentic Desktop 1.5 Type Library.

Most of the given examples are written in VisualBasic. These code snippets assume that there is a variable defined and set, called **objSpy of type Application**. There are also some code samples written in JavaScript.

3.2.1 Application

See also

Methods

GetDatabaseImportElementList
GetDatabaseSettings
GetDatabaseTables
ImportFromDatabase
CreateXMLSchemaFromDBStructure

GetTextImportElementList
GetTextImportExportSettings
ImportFromText

ImportFromWord

ImportFromSchema

GetExportSettings

NewProject
OpenProject

AddMacroMenuItem
ClearMacroMenu

ShowForm

ShowApplication

URLDelete
URLMakeDirectory

AddXSLT_XQParameter
GetXSLT_XQParameterCount
GetXSLT_XQParameterName
GetXSLT_XQParameterXPath
RemoveXSLT_XQParameter

FindInFiles

Quit

Properties

Application
Parent

ActiveDocument
Documents

CurrentProject

© 2015 Altova GmbH

Dialogs

WarningNumber
WarningText

Status
MajorVersion
MinorVersion
Edition
IsAPISupported
ServicePackVersion

Description
Application is the root for all other objects. It is the only object you can create by `CreateObject` (VisualBasic) or other similar COM related functions.

Example

```
Dim objSpy As Application
Set objSpy = CreateObject("XMLSpy.Application")
```

Events

OnBeforeOpenDocument

See also

Event: `OnBeforeOpenDocument(objDialog as FileSelectionDlg)`

Description
This event gets fired whenever a document gets opened via the OpenFile or OpenURL menu command. It is sent after a document file has been selected but before the document gets opened. The file selection dialog object is initialized with the name of the selected document file. You can modify this selection. To continue the opening of the document leave the `FileSelectionDlg.DialogAction` property of *io_objDialog* at its default value `spyDialogOK`. To abort the opening of the document set this property to `spyDialogCancel`.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeOpenDocument(objDialog)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeOpenDocument(objDialog)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (26, ...)  // nEventId = 26
```

OnBeforeOpenProject

See also

Event: `OnBeforeOpenProject(objDialog as FileSelectionDlg)`

Description
This event gets fired after a project file has been selected but before the project gets opened. The file selection dialog object is initialized with the name of the selected project file. You can modify this selection. To continue the opening of the project leave the `FileSelectionDlg.DialogAction` property of *io_objDialog* at its default value `spyDialogOK`. To abort the opening of the project set this property to `spyDialogCancel`.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
```
Function On_BeforeOpenProject(objDialog)
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeOpenProject(objDialog)
{
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (25, ...)  // nEventId = 25
```

OnDocumentOpened

See also

Event: `OnDocumentOpened(objDocument as Document)`

Description
This event gets fired whenever a document opens in Authentic Desktop. This can happen due to opening a file with the OpenFile or OpenURL dialog, creating a new file or dropping a file onto Authentic Desktop. The new document gets passed as parameter. The operation cannot be canceled.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:
```
Function On_OpenDocument(objDocument)
End Function
```

XMLSpy scripting environment - JScript:
```
function On_OpenDocument(objDocument)
```


© 2015 Altova GmbH

```
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (7, ...)   // nEventId = 7
```

OnProjectOpened

See also

Event: `OnProjectOpened(objProject as SpyProject)`

Description
This event gets fired whenever a project gets opened in Authentic Desktop. The new project gets passed as parameter.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:

```
Function On_OpenProject(objProject)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_OpenProject(objProject)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (6, ...)   // nEventId = 6
```

ActiveDocument

See also

Property: `ActiveDocument` as `Document`

Description
Reference to the active document. If no document is open, `ActiveDocument` is null (nothing).

Errors

1111	The application object is no longer valid.
1100	Invalid address for the return parameter was specified.

AddMacroMenuItem

See also

Method: `AddMacroMenuItem(strMacro as String,strDisplayText as String)`

Description
Adds a menu item to the **Tools** menu. This new menu item invokes the macro defined by `strMacro`. See also "Calling macros from Authentic Desktop".

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1108 Number of macro items is limited to 16 items.

AddXSLT_XQParameter

Method: `AddXSLT_XQParameter(name as String, XPath as String)`

Description
Adds an XSLT or XQuery parameter. The parameter's name and value are the two arguments of the method.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.
1124 The XPath expression is not set.
1125 Not a QName.
1126 The specified XPath is not valid. Reason for invalidity appended.
1127 A parameter with the submitted name already exists.

Application

See also

Property: `Application` as `Application` (read-only)

Description
Accesses the Authentic Desktop application object.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

ClearMacroMenu

See also

Method: `ClearMacroMenu()`

Return Value
None

Description

© 2015 Altova GmbH

Removes all menu items from the **Tools** menu. See also Running macros.

Errors
1111 The application object is no longer valid.

CreateXMLSchemaFromDBStructure

See also

Method: `CreateXMLSchemaFromDBStructure(pImportSettings as DatabaseConnection, pTables as ElementList)`

Description
`CreateXMLSchemaFromDBStructure` creates from a database specified in `pImportSettings` for the defined tables in `pTables` new XML Schema document(s) describing the database tables structure.

The parameter `pTables` specifies which table structures the XML Schema document should contain. This parameter can be NULL, specifying that all table structures will be exported.

See also GetDataBaseTables.

Errors
1112 Invalid database specified.
1120 Database import failed.

CurrentProject

See also

Property: `CurrentProject` as `SpyProject`

Description
Reference to the active document. If no project is open, `CurrentProject` is null (nothing).

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

Dialogs

See also

Property: `Dialogs` as `Dialogs` (read-only)

Description
Access the built-in dialogs of Authentic Desktop.

Errors
1111 The application object is no longer valid.

1100 Invalid address for the return parameter was specified.

Documents

See also

Property: `Documents` as `Documents`

Description
Collection of all open documents.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

Edition

See also

Property: `Edition` as String

Description
Returns the edition of the application, for example `Altova Authentic Desktop Enterprise Edition` for the Enterprise edition.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

FindInFiles

See also

Method: FindInFiles(*pSettings* as FindInFilesDlg) as FindInFilesResults

Description
Returns a FindInFilesResults object containing information about the files that matched the specified settings.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetDatabaseImportElementList

See also

Method: `GetDatabaseImportElementList(`*`pImportSettings`* as `DatabaseConnection)` as `ElementList`

Description
The function returns a collection of `ElementListItems` where the properties

 © 2015 Altova GmbH

ElementListItem.Name contain the names of the fields that can be selected for import and the properties ElementListItem.ElementKind are initialized either to *spyXMLDataAttr* or *spyXMLDataElement*, depending on the value passed in DatabaseConnection.AsAttributes. This list serves as a filter to what finally gets imported by a future call to ImportFromDatabase. Use ElementList.RemoveElement to exclude fields from import.

Properties mandatory to be filled out for the database connection are one of DatabaseConnection.File, DatabaseConnection.ADOConnection and DatabaseConnection.ODBCConnection, as well as DatabaseConnection.SQLSelect. Use the property DatabaseConnection.AsAttributes to initialize ElementListItem.ElementKind of the resulting element list to either *spyXMLDataAttr* or *spyXMLDataElement*, respectively.

Example
See example at ImportFromDatabase.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1112 Invalid database specified.
1114 Select statement is missing.
1119 database element list import failed.

GetDatabaseSettings

See also

Method: GetDatabaseSettings() as DatabaseConnection

Description
GetDatabaseSettings creates a new object of database settings. The object is used to specify database connection parameters for the methods GetDatabaseTables, GetDatabaseImportElementList, ImportFromDatabase, ImportFromSchema and ExportToDatabase.

Example
See example of ImportFromDatabase.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetDatabaseTables

See also

Method: GetDatabaseTables(*pImportSettings* as DatabaseConnection) as ElementList

Description

GetDatabaseTables reads the table names from the database specified in *pImportSettings*. Properties mandatory to be filled out for the database connection are one of DatabaseConnection.File, DatabaseConnection.ADOConnection and DatabaseConnection.ODBCConnection. All other properties are ignored.
The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of tables stored in the specified database. The remaining properties of ElementListItem are unused.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1112 Invalid database specified.
1113 Error while reading database table information.
1118 Database table query failed.

Example

```
Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings
objImpSettings.ADOConnection = TxtADO.Text

'store table names in list box
ListTables.Clear

Dim objList As ElementList
Dim objItem As ElementListItem
On Error GoTo ErrorHandler
Set objList = objSpy.GetDatabaseTables(objImpSettings)

For Each objItem In objList
 ListTables.AddItem objItem.Name
Next
```

GetExportSettings

See also

Method: GetExportSettings() as ExportSettings (read-only)

Description

GetExportSettings creates a new object of common export settings. This object is used to pass the parameters to the export functions and defines the behaviour of the export calls. See also the export functions from Document.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetTextImportElementList

See also

Method: GetTextImportElementList(pImportSettings as

© 2015 Altova GmbH

TextImportExportSettings) as ElementList

Description

GetTextImportElementList retrieves importing information about the text-file as specified in pImportSettings. The function returns a collection of ElementListItems where the properties ElementListItem.Name contain the names of the fields found in the file. The values of remaining properties are undefined.

If the text-file does not contain a column header, set pImportSettings.HeaderRow to False. The resulting element list will contain general column names like 'Field1' and so on.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1115 Error during text element list import. Cannot create parser for import file.
1116 Error during text element list import.

Example

```
' ------------------------------------------------------
' VBA client code fragment - import selected fields from text file
' ------------------------------------------------------
Dim objImpSettings As TextImportExportSettings
Set objImpSettings = objSpy.GetTextImportExportSettings

objImpSettings.ImportFile = "C:\ImportMe.txt"
objImpSettings.HeaderRow = False

Dim objList As ElementList
Set objList = objSpy.GetTextImportElementList(objImpSettings)

'exclude first column
objList.RemoveItem 1

Dim objImpDoc As Document
On Error Resume Next
Set objImpDoc = objSpy.ImportFromText(objImpSettings, objList)
CheckForError
```

GetTextImportExportSettings

See also

Method: GetTextImportExportSettings() as TextImportExportSettings (read-only)

Description

GetTextImportExportSettings creates a new object of common import and export settings for text files. See also the example for Application.GetTextImportElementList.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetXSLT_XQParameterCount

Method: `GetXSLT_XQParameterCount()` as Long

Description
Returns the number of XSLT and XQuery parameters.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetXSLT_XQParameterName

Method: `GetXSLT_XQParameterName(index as Long)` as String

Description
Returns the name of the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

GetXSLT_XQParameterXPath

Method: `GetXSLT_XQParameterXPath(index as Long)` as String

Description
Returns the XPath expression of the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

ImportFromDatabase

See also

Method: `ImportFromDatabase(pImportSettings as DatabaseConnection,pElementList as ElementList) as Document`

Return Value
Creates a new document containing the data imported from the database.

Description
`ImportFromDatabase` imports data from a database as specified in `pImportSettings` and creates a new document containing the data imported from the database. Properties mandatory to

© 2015 Altova GmbH

be filled out are one of DatabaseConnection.File, DatabaseConnection.ADOConnection or DatabaseConnection.ODBCConnection and DatabaseConnection.SQLSelect. Additionally, you can use DatabaseConnection.AsAttributes, DatabaseConnection.ExcludeKeys, DatabaseConnection.IncludeEmptyElements and NumberDateTimeFormat to further parameterize import.

The parameter pElementList specifies which fields of the selected data gets written into the newly created document, and which are created as elements and which as attributes. This parameter can be NULL, specifying that all selected fields will be imported as XML elements.

See GetDatabaseSettings and GetDatabaseImportElementList for necessary steps preceding any import of data from a database.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1112 Invalid database specified.
1114 Select statement is missing.
1117 Transformation to XML failed.
1120 Database import failed.

Example

```
Dim objImpSettings As DatabaseConnection
Set objImpSettings = objSpy.GetDatabaseSettings

objImpSettings.ADOConnection = strADOConnection
objImpSettings.SQLSelect = "SELECT * FROM MyTable"

Dim objDoc As Document
On Error Resume Next
Set objDoc = objSpy.ImportFromDatabase(objImpSettings,
objSpy.GetDatabaseImportElementList(objImpSettings))
' CheckForError here
```

ImportFromSchema

See also

Method: ImportFromSchema(*pImportSettings* as DatabaseConnection,*strTable* as String,*pSchemaDoc* as Document) as Document

Return Value

Creates a new document filled with data from the specified database as specified by the schema definition in *pSchemaDoc*.

Description

ImportFromSchema imports data from a database specified in pImportSettings. Properties mandatory to be filled out are one of DatabaseConnection.File, DatabaseConnection.ADOConnection or DatabaseConnection.ODBCConnection.

Additionally, you can use DatabaseConnection.AsAttributes, DatabaseConnection.ExcludeKeys and NumberDateTimeFormat to further parameterize import. All other properties get ignored.

ImportFromSchema does not use and explicit SQL statement to select the data. Instead, it expects a structure definition of the document to create in form of an XML schema document in *pSchemaDoc*. From this definition the database select statement is automatically deduced. Specify in *strTable* the table name of the import root that will become the root node in the new document.

See GetDatabaseSettings and GetDatabaseTables for necessary steps preceding an import from a database based on a schema definition. To create the schema definition file use command 'create database schema' from the 'convert' menu of Authentic Desktop.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from database failed.
1112 Invalid database specified.
1120 Database import failed.
1121 Could not create validator for the specified schema.
1122 Failed parsing schema for database import.

ImportFromText

See also

Method: ImportFromText(*pImportSettings* as TextImportExportSettings,*pElementList* as ElementList) as Document

Description

ImportFromText imports the text file as specified in pImportSettings. The parameter pElementList can be used as import filter. Either pass the list returned by a previous call to GetTextImportElementList or null to import all columns. To avoid import of unnecessary columns use ElementList.RemoveElement to remove the corresponding field names from pElementList before calling ImportFromText.
The method returns the newly created document containing the imported data. This document is the same as the active document of Authentic Desktop.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1107 Import from text file failed.
1117 Transformation to XML failed.

Example

```
' ------------------------------------------------------
' VBA client code fragment - import from text file
' ------------------------------------------------------
Dim objImpSettings As TextImportExportSettings
Set objImpSettings = objSpy.GetTextImportExportSettings

objImpSettings.ImportFile = strFileName
```


© 2015 Altova GmbH

```
objImpSettings.HeaderRow = False

Dim objImpDoc As Document
On Error Resume Next
Set objImpDoc = objSpy.ImportFromText(objImpSettings,
  objSpy.GetTextImportElementList(objImpSettings))

CheckForError
```

ImportFromWord

See also

Method: `ImportFromWord(strFile` as String) as `Document`

Description
`ImportFromWord` imports the MS-Word Document `strFile` into a new XML document.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified. Import from document failed.

IsAPISupported

See also

Property: `IsAPISupported` as Boolean

Description
Returns whether the API is supported in this version or not.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

MajorVersion

See also

Property: `MajorVersion` as Integer

Description
Returns the application version's major number, for example `15` for 2013 versions, and `16` for 2014 versions..

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

MinorVersion

See also

Property: `MinorVersion` as Integer

Description
Returns the application version's minor number.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

NewProject

See also

Method: `NewProject(strPath` as String,`bDiscardCurrent` as Boolean`)`

Description
`NewProject` creates a new project.

If there is already a project open that has been modified and `bDiscardCurrent` is false, then `NewProject()` fails.

Errors
1111 The application object is no longer valid.
1102 A project is already open but *bDiscardCurrent* is *true*.
1103 Creation of new project failed.

OpenProject

See also

Method: `OpenProject(strPath` as String,`bDiscardCurrent` as Boolean,`bDialog` as Boolean`)`

Parameters
`strPath`
Path and file name of the project to open. Can be empty if `bDialog` is true.

`bDiscardCurrent`
Discard currently open project and possible lose changes.

`bDialog`
Show dialogs for user input.

Return Value
None

Description

© 2015 Altova GmbH

`OpenProject` opens an existing project. If there is already a project open that has been modified and `bDiscardCurrent` is false, then `OpenProject()` fails.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.
1101 Cannot open specified project.
1102 A project is already open but *bDiscardCurrent* is *true*.

Parent

See also

Property: `Parent` as `Application` (read-only)

Description
Accesses the Authentic Desktop application object.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

Quit

See also

Method: Quit()

Return Value
None

Description
This method terminates Authentic Desktop. All modified documents will be closed without saving the changes. This is also true for an open project.

If Authentic Desktop was automatically started as an automation server by a client program, the application will not shut down automatically when your client program shuts down if a project or any document is still open. Use the Quit method to ensure automatic shut-down.

Errors

1111 The application object is no longer valid.

ReloadSettings

See also

Method: `ReloadSettings`

Return Value

Description
The application settings are reloaded from the registry.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.

RemoveXSLT_XQParameter

Method: `RemoveXSLT_XQParameter(index as Long)`

Description
Removes the XSLT or XQuery parameter identified by the supplied index.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

RunMacro

See also

Method: `RunMacro(strMacro as String)`

Return Value

Description
Calls the specified macro either from the project scripts (if present) or from the global scripts.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.

ScriptingEnvironment

See also

Property: `ScriptingEnvironment` as IUnknown (read-only)

Description
Reference to any active scripting environment. This property makes it possible to access the TypeLibrary of the XMLSpyFormEditor.exe application which is used as the current scripting environment.

Available with TypeLibrary version 1.5

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

ServicePackVersion

See also

Property: `ServicePackVersion` as Long

Description
Returns the Service Pack version number of the application. Eg: 1 for 2010 R2 SP1

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

ShowApplication

See also

Method: `ShowApplication(bShow` as Boolean)

Return Value
None

Description
The method shows `(bShow = True)` or hides `(bShow = False)` Authentic Desktop.

Errors
1110 The application object is no longer valid.

ShowFindInFiles

See also

Method: ShowFindInFiles `(pSettings as FindInFilesDlg)` as Boolean

Return Value
Returns false if the user pressed the Cancel button, true otherwise.

Description
Displays the FindInFiles dialog preset with the given settings. The user modifications of the settings are stored in the passed dialog object.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.

ShowForm

See also

Method: `ShowForm(strFormName` as String) as Long

Return Value

Returns zero if the user pressed a Cancel button or the form calls `TheView.Cancel()`.

Description
Displays the form `strFormName`.

Forms, event handlers and macros can be created with the Scripting Environment. Select "Switch to scripting environment" from the **Tools** menu to invoke the Scripting Environment.

Errors
1111 The application object is no longer valid.
1100 Invalid parameter or invalid address for the return parameter was specified.

Status

See also

Property: `Status` as ENUMApplicationStatus

Description
Returns the current status of the running application.

Errors
1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

URLDelete

See also

Method: `URLDelete(strURL` as String, `strUser` as String, `strPassword` as String)

Return Value
None

Description
The method deletes the file at the URL `strURL`.

Errors
1111 The application object is no longer valid.
1109 Error deleting file at specified URL.

URLMakeDirectory

See also

Method: `URLMakeDirectory(strURL` as String, `strUser` as String, `strPassword` as String)

Return Value
None

Description

© 2015 Altova GmbH

The method creates a new directory at the URL `strURL`.

Errors

1111 The application object is no longer valid.
1100 Invalid parameter specified.

Visible

See also

Property: `Visible` as `VARIANT_BOOL`

Description

Sets or gets the visibility attribute of Authentic Desktop. This standard automation property makes usage of `ShowApplication` obsolete.

Errors

1110 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

WarningNumber

See also

Property: `WarningNumber` as integer

Description

Some methods fill the property `WarningNumber` with additional information if an error occurs.

Currently just `Documents.OpenFile` fills this property.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

WarningText

See also

Property: `WarningText` as String

Description

Some methods fill the property `WarningText` with additional information if an error occurs.

Currently just `Documents.OpenFile` fills this property.

Errors

1111 The application object is no longer valid.
1100 Invalid address for the return parameter was specified.

3.2.2 AuthenticContextMenu

The context menu interface provides the mean for the user to customize the context menus shown in Authentic. The interface has the methods listed in this section.

CountItems

Method: `CountItems()` nItems as long

Return Value
Returns the number of menu items.

Errors
2501 Invalid object.

DeleteItem

Method: `DeleteItem(IndexPosition as long)`

Return Value
Deletes the menu item that has the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

GetItemText

Method: `GetItemText(IndexPosition as long)` MenuItemName as string

Return Value
Gets the name of the menu item located at the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

InsertItem

Method: `InsertItem(IndexPosition as long, MenuItemName as string, MacroName as string)`

Return Value
Inserts a user-defined menu item at the position in the menu specified in the first parameter and having the name submitted in the second parameter. The menu item will start a macro, so a valid macro name must be submitted.

Errors
2501 Invalid object
2502 Invalid index

© 2015 Altova GmbH

2503 No such macro
2504 Internal error

SetItemText

Method: `SetItemText(IndexPosition as long, MenuItemName as string)`

Return Value
Sets the name of the menu item located at the index position submitted in the first parameter.

Errors
2501 Invalid object
2502 Invalid index

3.2.3 AuthenticDataTransfer

Renamed from DocEditDataTransfer to AuthenticDataTransfer

The `DocEditView` object is renamed to `OldAuthenticView`.
`DocEditSelection` is renamed to `AuthenticSelection`.
`DocEditEvent` is renamed to `AuthenticEvent`.
`DocEditDataTransfer` is renamed to `AuthenticDataTransfer`.

Their usage—except for `AuthenticDataTransfer`—is no longer recommended. We will continue to support existing functionality for a yet undefined period of time but no new features will be added to these interfaces.

For examples on migrating from DocEdit to Authentic see the description of the different methods and properties of the different DocEdit objects.

See also

Methods

`getData`

Properties

`dropEffect`
`ownDrag`
`type`

Description

The events `OnDragOver` and `OnBeforeDrop` provide information about the object being dragged with an instance of type `AuthenticDataTransfer`. It contains a description of the dragged object and its content. The latter is available either as string or a pointer to a COM object supporting the `IUnkown` interface.

dropEffect

See also

Property: `dropEffect` as long

Description
The property stores the drop effect from the default event handler. You can set the drop effect if you change this value and return TRUE for the event handler.

Errors
2101 Invalid address for the return parameter was specified.

getData

See also

© 2015 Altova GmbH

Method: `getData()` as Variant

Description
Retrieve the data associated with the dragged object. Depending on `AuthenticDataTransfer.type`, that data is either a string or a COM interface pointer of type `IUnknown`.

Errors
2101 Invalid address for the return parameter was specified.

ownDrag

See also

Property: `ownDrag as` Boolean (read-only)

Description
The property is `TRUE` if the current dragging source comes from inside vista Authentic.

Errors
2101 Invalid address for the return parameter was specified.

type

See also

Property: `type` as String (read-only)

Description
Holds the type of data you get with the `DocEditDataTransfer.getData` method.

Currently supported data types are:

OWN	data from vista Authentic itself
TEXT	plain text
UNICODETEXT	plain text as UNICODE

Errors
2101 Invalid address for the return parameter was specified.

3.2.4 AuthenticEventContext

The `EventContext` interface gives access to many properties of the context in which a macro is executed.

EvaluateXPath

Method: `EvaluateXPath` (`strExpression as string`) as strValue as string

Return Value
The method evaluates the XPath expression in the context of the node within which the event was triggered and returns a string.

Description
`EvaluateXPath()` executes an XPath expressions with the given event context. The result is returned as string, in the case of a sequence it is a space-separated string.

Errors

2201	Invalid object.
2202	No context.
2209	Invalid parameter.
2210	Internal error.
2211	XPath error.

GetEventContextType

Method: `GetEventContextType ()` Type as `AuthenticEventContextType` enumeration

Return Value
Returns the context node type.

Description
`GetEventContextType` allows the user to determine whether the macro is in an XML node or in an XPath atomic item context. The enumeration `AuthenticEventContextType` is defined as follows:

```
authenticEventContextXML,
authenticEventContextAtomicItem,
authenticEventContextOther
```

If the context is a normal XML node, the `GetXMLNode()` function gives access to it (returns `NULL` if not).

Errors

2201	Invalid object.
2202	No context.
2209	Invalid parameter.

GetNormalizedTextValue

Method: `GetNormalizedTextValue()` strValue as string

Return Value

© 2015 Altova GmbH

Returns the value of the current node as string

Errors

2201 Invalid object.
2202 No context.
2203 Invalid context
2209 Invalid parameter.

GetVariableValue

Method: `GetVariableValue(strName as string) strValue as string`

Return Value
Gets the value of the variable submitted as the parameter.

Description
`GetVariableValue` gets the variable's value in the scope of the context.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
	AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
}
```

Errors

2201 Invalid object.
2202 No context.
2204 No such variable in scope
2205 Variable cannot be evaluated
2206 Variable returns sequence
2209 Invalid parameter

GetXMLNode

Method: `GetXMLNode()` Node as XMLData object

Return Value
Returns the context XML node or `NULL`

Errors

2201 Invalid object.
2202 No context.
2203 Invalid context
2209 Invalid parameter.

IsAvailable

Method: `IsAvailable()` as Boolean

Return Value
Returns true if `EventContext` is set, false otherwise.

Errors

2201 Invalid object.

SetVariableValue

Method: `SetVariableValue(strName as string, strValue as string)`

Return Value

Sets the value (second parameter) of the variable submitted in the first parameter.

Description

`SetVariableValue` sets the variable's value in the scope of the context.

```
nZoom = parseInt( AuthenticView.EventContext.GetVariableValue( 'Zoom' ) );
if ( nZoom > 1 )
{
	AuthenticView.EventContext.SetVariableValue( 'Zoom', nZoom - 1 );
}
```

Errors

2201 Invalid object.
2202 No context.
2204 No such variable in scope
2205 Variable cannot be evaluated
2206 Variable returns sequence
2207 Variable read-only
2208 No modification allowed

© 2015 Altova GmbH

3.2.5 AuthenticRange

See also

The first table lists the properties and methods of AuthenticRange that can be used to navigate through the document and select specific portions.

Properties		Methods
Application	Clone	MoveBegin
FirstTextPosition	CollapsToBegin	MoveEnd
FirstXMLData	CollapsToEnd	NextCursorPosition
FirstXMLDataOffset	ExpandTo	PreviousCursorPosition
LastTextPosition	Goto	Select
LastXMLData	GotoNext	SelectNext
LastXMLDataOffset	GotoPrevious	SelectPrevious
Parent	IsEmpty	SetFromRange
	IsEqual	

The following table lists the content modification methods, most of which can be found on the right/button mouse menu.

Properties	Edit operations	Dynamic table operations
Text	Copy	AppendRow
	Cut	DeleteRow
	Delete	DuplicateRow
	IsCopyEnabled	InsertRow
	IsCutEnabled	IsFirstRow
	IsDeleteEnabled	IsInDynamicTable
	IsPasteEnabled	IsLastRow
	Paste	MoveRowDown
		MoveRowUp

The following methods provide the functionality of the Authentic entry helper windows for range objects.

Operations of the entry helper windows

Elements	Attributes	Entities
CanPerformActionWith	GetElementAttributeValue	GetEntityNames
CanPerformAction	GetElementAttributeNames	InsertEntity
PerformAction	GetElementHierarchy	
	HasElementAttribute	
	IsTextStateApplied	
	SetElementAttributeValue	

Description

AuthenticRange objects are the 'cursor' selections of the automation interface. You can use them to point to any cursor position in the Authentic view, or select a portion of the document. The operations available for AuthenticRange objects then work on this selection in the same way, as the corresponding operations of the user interface do with the current user interface selection. The main difference is that you can use an arbitrary number of AuthenticRange

objects at the same time, whereas there is exactly one cursor selection in the user interface.

To get to an initial range object use `AuthenticView.Selection`, to obtain a range corresponding with the current cursor selection in the user interface. Alternatively, some trivial ranges are accessible via the read/only properties `AuthenticView.DocumentBegin`, `AuthenticView.DocumentEnd`, and `AuthenticView.WholeDocument`. The most flexible method is `AuthenticView.Goto`, which allows navigation to a specific portion of the document within one call. For more complex selections, combine the above, with the various navigation methods on range objects listed in the first table on this page.

Another method to select a portion of the document is to use the position properties of the range object. Two positioning systems are available and can be combined arbitrarily:

- **Absolute** text cursor positions, starting with position 0 at the document beginning, can be set and retrieved for the beginning and end of a range. For more information see `FirstTextPosition` and `LastTextPosition`. This method requires complex internal calculations and should be used with care.
- The **XMLData** element and a text position inside this element, can be set and retrieved for the beginning and end of a range. For more information see `FirstXMLData`, `FirstXMLDataOffset`, `LastXMLData`, and `LastXMLDataOffset`. This method is very efficient but requires knowledge on the underlying document structure. It can be used to locate XMLData objects and perform operations on them otherwise not accessible through the user interface.

Modifications to the document content can be achieved by various methods:

- The `Text` property allows you to retrieve the document text selected by the range object. If set, the selected document text gets replaced with the new text.
- The standard document edit functions `Cut`, `Copy`, `Paste` and `Delete`.
- Table operations for tables that can grow dynamically.
- Methods that map the functionality of the Authentic entry helper windows.
- Access to the `XMLData` objects of the underlying document to modify them directly.

AppendRow

See also

Method: `AppendRow()` as Boolean

Description
If the beginning of the range is inside a dynamic table, this method inserts a new row at the end of the selected table. The selection of the range is modified to point to the beginning of the new row. The function returns *true* if the append operation was successful, otherwise *false*.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Append row at end of current dynamically growable table
' ----------------------------------------------------------
```

 © 2015 Altova GmbH

```
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
     objRange.AppendRow
     ' objRange points to beginning of new row
     objRange.Select
End If
```

Application

See also

Property: `Application` as `Application` (read-only)

Description
Accesses the Authentic Desktop application object.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

CanPerformAction

See also

Method: `CanPerformAction` (*`eAction`* as `SPYAuthenticActions`, *`strElementName`* as String) as Boolean

Description
`CanPerformAction` and its related methods enable access to the entry-helper functions of Authentic. This function allows easy and consistent modification of the document content, without having to know exactly where the modification will take place. The beginning of the range object is used to locate the next valid location where the specified action can be performed. If the location can be found, the method returns *True*, otherwise it returns *False*.

HINT: To find out all valid element names for a given action, use `CanPerformActionWith`.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.
2007	Invalid action was specified.

Examples
See `PerformAction`.

CanPerformActionWith

See also

Method: `CanPerformActionWith` (*`eAction`* as `SPYAuthenticActions`,

out_arrElementNames as Variant)

Description
PerformActionWith and its related methods, enable access to the entry-helper functions of Authentic. These function allows easy and consistent modification of the document content without having to know exactly where the modification will take place.

This method returns an array of those element names that the specified action can be performed with.

HINT: To apply the action use CanPerformActionWith.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.
2007 Invalid action was specified.

Examples
See PerformAction.

Clone

See also

Method: Clone() as AuthenticRange

Description
Returns a copy of the range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

CollapsToBegin

See also

Method: CollapsToBegin() as AuthenticRange

Description
Sets the end of the range object to its begin. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

CollapsToEnd

See also

Method: CollapsToEnd() as AuthenticRange

© 2015 Altova GmbH

Description
Sets the beginning of the range object to its end. The method returns the modified range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Copy

See also

Method: `Copy()` as Boolean

Description
Returns *False* if the range contains no portions of the document that may be copied.
Returns *True* if text, and in case of fully selected XML elements the elements as well, has been copied to the copy/paste buffer.

Errors
2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Cut

See also

Method: `Cut()` as Boolean

Description
Returns *False* if the range contains portions of the document that may not be deleted.
Returns *True* after text, and in case of fully selected XML elements the elements as well, has been deleted from the document and saved in the copy/paste buffer.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Delete

See also

Method: `Delete()` as Boolean

Description
Returns *False* if the range contains portions of the document that may not be deleted.
Returns *True* after text, and in case of fully selected XML elements the elements as well, has been deleted from the document.

Errors
2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

DeleteRow

See also

Method: `DeleteRow()` as Boolean

Description
If the beginning of the range is inside a dynamic table, this method deletes the selected row. The selection of the range gets modified to point to the next element after the deleted row. The function returns *true,* if the delete operation was successful, otherwise *false*.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Delete selected row from dynamically growing table
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we are in a table
If objRange.IsInDynamicTable Then
     objRange.DeleteRow
End If
```

DuplicateRow

See also

Method: `DuplicateRow()` as Boolean

Description
If the beginning of the range is inside a dynamic table, this method inserts a duplicate of the current row after the selected one. The selection of the range gets modified to point to the beginning of the new row. The function returns *true* if the duplicate operation was successful, otherwise *false*.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' duplicate row in current dynamically growable table
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection
```


© 2015 Altova GmbH

```
' check if we can insert something
If objRange.IsInDynamicTable Then
     objRange.DuplicateRow
     ' objRange points to beginning of new row
     objRange.Select
End If
```

EvaluateXPath

Method: `EvaluateXPath` (strExpression as string) strValue as string

Return Value
The method returns a string

Description
`EvaluateXPath()` executes an XPath expressions with the context node being the beginning of the range selection. The result is returned as string, in the case of a sequence it is a space-separated string. If XML context node is irrelevant, the user may provide any node, like `AuthenticView.XMLDataRoot`.

Errors

2001	Invalid object
2005	Invalid parameter
2008	Internal error
2202	Missing context node
2211	XPath error

ExpandTo

See also

Method: `ExpandTo` (*`eKind`* as `SPYAuthenticElementKind`), as `AuthenticRange`

Description
Selects the whole element of type `eKind`, that starts at, or contains, the first cursor position of the range. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Range expansion would be beyond end of document.
2005	Invalid address for the return parameter was specified.

FirstTextPosition

See also

Property: `FirstTextPosition` as Long

Description
Set or get the left-most text position index of the range object. This index is always less or equal to `LastTextPosition`. Indexing starts with 0 at document beginning, and increments with every different position that the text cursor can occupy. Incrementing the test position by 1, has the same effect as the cursor-right key. Decrementing the test position by 1 has the same effect

as the cursor-left key.

If you set FirstTextPosition to a value greater than the current LastTextPosition, LastTextPosition gets set to the new FirstTextPosition.

HINT: Use text cursor positions with care, since this is a costly operation compared to XMLData based cursor positioning.

Errors

2001 The authentic range object, or its related view object is not valid.
2005 Invalid address for the return parameter was specified.
2006 A text position outside the document was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
Dim objRange
' we need to get a (any) range object first
Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.isEqual(objAuthenticView.WholeDocument) Then
     MsgBox "Test using direct text cursor positioning was ok"
Else
     MsgBox "Ooops!"
End If
```

FirstXMLData

See also

Property: FirstXMLData as XMLData

Description

Set or get the first XMLData element in the underlying document that is partially, or completely selected by the range. The exact beginning of the selection is defined by the FirstXMLDataOffset attribute.

Whenever you set FirstXMLData to a new data object, FirstXMLDataOffset gets set to the first cursor position inside this element. Only XMLData objects that have a cursor position may be used. If you set FirstXMLData / FirstXMLDataOffset selects a position greater then the current LastXMLData / LastXMLDataOffset, the latter gets moved to the new start position.

 © 2015 Altova GmbH

HINT: You can use the FirstXMLData and LastXMLData properties, to directly access and manipulate the underlying XML document in those cases where the methods available with the AuthenticRange object are not sufficient.

Errors

2001 The authentic range object, or its related view object is not valid.
2005 Invalid address for the return parameter was specified.
2008 Internal error
2009 The XMLData object cannot be accessed.

Examples

```
' ------------------------------------------------
' Scripting environment - VBScript
' show name of currently selected XMLData element
' ------------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objXmlData
Set objXMLData = objAuthenticView.Selection.FirstXMLData
' authentic view adds a 'text' child element to elements
' of the document which have content. So we have to go one
' element up.
Set objXMLData = objXMLData.Parent
MsgBox "Current selection selects element " & objXMLData.Name
```

FirstXMLDataOffset

See also

Property: FirstXMLDataOffset as Long

Description

Set or get the cursor position offset inside FirstXMLData element for the beginning of the range. Offset positions are based on the characters returned by the Text property, and start with 0. When setting a new offset, use -1 to set the offset to the last possible position in the element. The following cases require specific attention:

- The textual form of entries in Combo Boxes, Check Boxes and similar controls can be different from what you see on screen. Although the data offset is based on this text, there only two valid offset positions, one at the beginning and one at the end of the entry. An attempt to set the offset to somewhere in the middle of the entry, will result in the offset being set to the end.
- The textual form of XML Entities might differ in length from their representation on the screen. The offset is based on this textual form.

If FirstXMLData / FirstXMLDataOffset selects a position after the current LastXMLData / LastXMLDataOffset, the latter gets moved to the new start position.

Errors

2001 The authentic range object, or its related view object is not valid.
2005 Invalid offset was specified.

Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ---------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
     objRange.Select()
Else
     MsgBox "Oops"
End If
```

GetElementAttributeNames

See also

Method: GetElementAttributeNames (*strElementName* as String, *out_arrAttributeNames* as Variant)

Description

Retrieve the names of all attributes for the enclosing element with the specified name. Use the element/attribute pairs, to set or get the attribute value with the methods GetElementAttributeValue and SetElementAttributeValue.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid address for the return parameter was specified.

Examples

See SetElementAttributeValue.

© 2015 Altova GmbH

GetElementAttributeValue

See also

Method: GetElementAttributeValue (*strElementName* as String, *strAttributeName* as String) as String

Description
Retrieve the value of the attribute specified in strAttributeName, for the element identified with strElementName. If the attribute is supported but has no value assigned, the empty string is returned. To find out the names of attributes supported by an element, use GetElementAttributeNames, or HasElementAttribute.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid element name was specified.
	Invalid attribute name was specified.
	Invalid address for the return parameter was specified.

Examples
See SetElementAttributeValue.

GetElementHierarchy

See also

Method: GetElementHierarchy (*out_arrElementNames* as Variant)

Description
Retrieve the names of all XML elements that are parents of the current selection. Inner elements get listed before enclosing elements. An empty list is returned whenever the current selection is not inside a single XMLData element.

The names of the element hierarchy, together with the range object uniquely identify XMLData elements in the document. The attributes of these elements can be directly accessed by GetElementAttributeNames, and related methods.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

GetEntityNames

See also

Method: GetEntityNames (*out_arrEntityNames* as Variant)

Description
Retrieve the names of all defined entities. The list of retrieved entities is independent of the current selection, or location. Use one of these names with the InsertEntity function.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples

See: GetElementHierarchy and InsertEntity.

GetVariableValue

Method: GetVariableValue(strName as string) strVal as string

Return Value

Gets the value of the variable named as the method's parameter.

Errors

2001 Invalid object.
2202 No context.
2204 No such variable in scope
2205 Variable cannot be evaluated
2206 Variable returns sequence
2209 Invalid parameter

Goto

See also

Method: Goto (*eKind* as SPYAuthenticElementKind, *nCount* as Long, *eFrom* as SPYAuthenticDocumentPosition) as AuthenticRange

Description

Sets the range to point to the beginning of the nCount element of type eKind. The start position is defined by the parameter eFrom.

Use positive values for nCount to navigate to the document end. Use negative values to navigate to the beginning of the document. The method returns the modified range object.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.
2004 Target lies before begin of document.
2005 Invalid element kind specified.
Invalid start position specified.
Invalid address for the return parameter was specified.

GotoNext

See also

Method: GotoNext (*eKind* as SPYAuthenticElementKind) as AuthenticRange

© 2015 Altova GmbH

Description
Sets the range to the beginning of the next element of type `eKind`. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

Examples

```
' ------------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
' ------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
     objRange.GotoNext(spyAuthenticWord).Select
     If ((Err.number - vbObjecterror) = 2003) Then
            bEndOfDocument = True
            Err.Clear
     ElseIf (Err.number <> 0) Then
            Err.Raise  ' forward error
     End If
Wend
```

GotoNextCursorPosition

See also

Method: `GotoNextCursorPosition()` as `AuthenticRange`

Description
Sets the range to the next cursor position after its current end position. Returns the modified object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2005	Invalid address for the return parameter was specified.

GotoPrevious

See also

Method: GotoPrevious (*eKind* as SPYAuthenticElementKind) as AuthenticRange

Description
Sets the range to the beginning of the element of type eKind which is before the beginning of the current range. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2004	Target lies before beginning of document.
2005	Invalid element kind specified. Invalid address for the return parameter was specified.

Examples

```
' ------------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
' ------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bEndOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
     objRange.GotoPrevious(spyAuthenticTag).Select
     If ((Err.number - vbObjecterror) = 2004) Then
            bBeginOfDocument = True
            Err.Clear
     ElseIf (Err.number <> 0) Then
            Err.Raise  ' forward error
     End If
Wend
```

GotoPreviousCursorPosition

See also

Method: GotoPreviousCursorPosition() as AuthenticRange

Description
Set the range to the cursor position immediately before the current position. Returns the modified object.

Errors

2001	The authentic range object, or its related view object is no longer valid.

© 2015 Altova GmbH

2004 Target lies before begin of document.
2005 Invalid address for the return parameter was specified.

HasElementAttribute

See also

Method: `HasElementAttribute` (*`strElementName`* as String, *`strAttributeName`* as String) as Boolean

Description
Tests if the enclosing element with name `strElementName`, supports the attribute specified in `strAttributeName`.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid element name was specified.
Invalid address for the return parameter was specified.

InsertEntity

See also

Method: `InsertEntity` (*`strEntityName`* as String)

Description
Replace the ranges selection with the specified entity. The specified entity must be one of the entity names returned by `GetEntityNames`.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Unknown entry name was specified.

Examples

```
' ----------------------------------------------------------
' Scripting environment - VBScript
' Insert the first entity in the list of available entities
' ----------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we get the names of all available entities as they
' are shown in the entry helper of XMLSpy
Dim arrEntities
objRange.GetEntityNames arrEntities

' we insert the first one of the list
If UBound(arrEntities) >= 0 Then
     objRange.InsertEntity arrEntities(0)
Else
     MsgBox "Sorry, no entities are available for this document"
End If
```

InsertRow

See also

Method: `InsertRow()` as Boolean

Description
If the beginning of the range is inside a dynamic table, this method inserts a new row before the current one. The selection of the range, gets modified to point to the beginning of the newly inserted row. The function returns *true* if the insert operation was successful, otherwise *false*.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

Examples

```
' ------------------------------------------------------------
' Scripting environment - VBScript
' Insert row at beginning of current dynamically growing table
' ------------------------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' check if we can insert something
If objRange.IsInDynamicTable Then
     objRange.InsertRow
     ' objRange points to beginning of new row
     objRange.Select
End If
```

IsCopyEnabled

See also

Property: `IsCopyEnabled` as Boolean (read-only)

Description
Checks if the copy operation is supported for this range.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

IsCutEnabled

See also

Property: `IsCutEnabled` as Boolean (read-only)

Description
Checks if the cut operation is supported for this range.

© 2015 Altova GmbH

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsDeleteEnabled

See also

Property: `IsDeleteEnabled` as Boolean (read-only)

Description
Checks if the delete operation is supported for this range.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsEmpty

See also

Method: `IsEmpty()` as Boolean

Description
Tests if the first and last position of the range are equal.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsEqual

See also

Method: `IsEqual` (*`objCmpRange`* as `AuthenticRange`) as Boolean

Description
Tests if the start and end of both ranges are the same.

Errors

2001 One of the two range objects being compared, is invalid.
2005 Invalid address for a return parameter was specified.

IsFirstRow

See also

Property: `IsFirstRow` as Boolean (read-only)

Description
Test if the range is in the first row of a table. Which table is taken into consideration depends on the extend of the range. If the selection exceeds a single row of a table, the check is if this table is the first element in an embedding table. See the entry helpers of the user manual for more

information.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsInDynamicTable

See also

Method: `IsInDynamicTable()` as Boolean

Description
Test if the whole range is inside a table that supports the different row operations like 'insert', 'append', duplicate, etc.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsLastRow

See also

Property: `IsLastRow` as Boolean (read-only)

Description
Test if the range is in the last row of a table. Which table is taken into consideration depends on the extend of the range. If the selection exceeds a single row of a table, the check is if this table is the last element in an embedding table. See the entry helpers of the user manual for more information.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsPasteEnabled

See also

Property: `IsPasteEnabled` as Boolean (read-only)

Description
Checks if the paste operation is supported for this range.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

IsSelected

Property: `IsSelected` as Boolean

Description
Returns true() if selection is present. The selection range still can be empty: that happens when e.g. only the cursor is set.

IsTextStateApplied

See also

Method: `IsTextStateApplied` (*`i_strElementName`* as String) as Boolean

Description
Checks if all the selected text is embedded into an XML Element with name `i_strElementName`. Common examples for the parameter `i_strElementName` are "strong", "bold" or "italic".

Errors

2001	The authentic range object, or its related view object is no longer valid.
2005	Invalid address for the return parameter was specified.

LastTextPosition

See also

Property: `LastTextPosition` as Long

Description
Set or get the rightmost text position index of the range object. This index is always greater or equal to `FirstTextPosition`. Indexing starts with 0 at the document beginning, and increments with every different position that the text cursor can occupy. Incrementing the test position by 1, has the same effect as the cursor-right key. Decreasing the test position by 1 has the same effect as the cursor-left key.

If you set `LastTextPosition` to a value less then the current `FirstTextPosition`, `FirstTextPosition` gets set to the new `LastTextPosition`.

HINT: Use text cursor positions with care, since this is a costly operation compared to XMLData based cursor positioning.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid address for the return parameter was specified.
2006	A text position outside the document was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
```

```
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

nDocStartPosition = objAuthenticView.DocumentBegin.FirstTextPosition
nDocEndPosition = objAuthenticView.DocumentEnd.FirstTextPosition

' let's create a range that selects the whole document
' in an inefficient way
Dim objRange
' we need to get a (any) range object first
Set objRange = objAuthenticView.DocumentBegin
objRange.FirstTextPosition = nDocStartPosition
objRange.LastTextPosition = nDocEndPosition

' let's check if we got it right
If objRange.isEqual(objAuthenticView.WholeDocument) Then
     MsgBox "Test using direct text cursor positioning was ok"
Else
     MsgBox "Oops!"
End If
```

LastXMLData

See also

Property: `LastXMLData` as `XMLData`

Description

Set or get the last `XMLData` element in the underlying document that is partially or completely selected by the range. The exact end of the selection is defined by the `LastXMLDataOffset` attribute.

Whenever you set `LastXMLData` to a new data object, `LastXMLDataOffset` gets set to the last cursor position inside this element. Only `XMLData` objects that have a cursor position may be used. If you set `LastXMLData` / `LastXMLDataOffset`, select a position less then the current `FirstXMLData` / `FirstXMLDataOffset`, the latter gets moved to the new end position.

HINT: You can use the `FirstXMLData` and `LastXMLData` properties to directly access and manipulate the underlying XML document in those cases, where the methods available with the `AuthenticRange` object are not sufficient.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid address for the return parameter was specified.
2008	Internal error
2009	The `XMLData` object cannot be accessed.

© 2015 Altova GmbH

LastXMLDataOffset

See also

Property: LastXMLDataOffset as Long

Description
Set or get the cursor position inside LastXMLData element for the end of the range.

Offset positions are based on the characters returned by the Text property and start with 0. When setting a new offset, use -1 to set the offset to the last possible position in the element. The following cases require specific attention:

- The textual form of entries in Combo Boxes, Check Boxes and similar controls can be different from what you see on the screen. Although, the data offset is based on this text, there only two valid offset positions, one at the beginning and one at the end of the entry. An attempt to set the offset to somewhere in the middle of the entry, will result in the offset being set to the end.
- The textual form of XML Entities might differ in length from their representation on the screen. The offset is based on this textual form.

If LastXMLData / LastXMLDataOffset selects a position before FirstXMLData / FirstXMLDataOffset, the latter gets moved to the new end position.

Errors

2001	The authentic range object, or its related view object is not valid.
2005	Invalid offset was specified. Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------------
' Scripting environment - VBScript
' Select the complete text of an XMLData element
' using XMLData based selection and ExpandTo
' ---------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' first we use the XMLData based range properties
' to select all text of the first XMLData element
' in the current selection
Dim objRange
Set objRange = objAuthenticView.Selection
objRange.FirstXMLDataOffset = 0  ' start at beginning of element text
objRange.LastXMLData = objRange.FirstXMLData  ' select only one element
objRange.LastXMLDataOffset = -1  ' select till its end

' the same can be achieved with the ExpandTo method
Dim objRange2
Set objRange2 = objAuthenticView.Selection.ExpandTo(spyAuthenticTag)

' were we successful?
If objRange.IsEqual(objRange2) Then
```

```
      objRange.Select()
Else
      MsgBox "Ooops"
End If
```

MoveBegin

See also

Method: MoveBegin (*eKind* as SPYAuthenticElementKind, *nCount* as Long) as AuthenticRange

Description
Move the beginning of the range to the beginning of the nCount element of type eKind. Counting starts at the current beginning of the range object.

Use positive numbers for nCount to move towards the document end, use negative numbers to move towards document beginning. The end of the range stays unmoved, unless the new beginning would be larger than it. In this case, the end is moved to the new beginning. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2004	Target lies before beginning of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

MoveEnd

See also

Method: MoveEnd (*eKind* as SPYAuthenticElementKind, *nCount* as Long) as AuthenticRange

Description
Move the end of the range to the begin of the nCount element of type eKind. Counting starts at the current end of the range object.

Use positive numbers for nCount to move towards the document end, use negative numbers to move towards document beginning. The beginning of the range stays unmoved, unless the new end would be less than it. In this case, the beginning gets moved to the new end. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2003	Target lies after end of document.
2004	Target lies before begin of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

© 2015 Altova GmbH

MoveRowDown

See also

Method: `MoveRowDown()` as Boolean

Description
If the beginning of the range is inside a dynamic table and selects a row which is not the last row in this table, this method swaps this row with the row immediately below. The selection of the range moves with the row, but does not otherwise change. The function returns *true* if the move operation was successful, otherwise *false*.

Errors
2001 The authentic range object or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

MoveRowUp

See also

Method: `MoveRowUp()` as Boolean

Description
If the beginning of the range is inside a dynamic table and selects a row which is not the first row in this table, this method swaps this row with the row above. The selection of the range moves with the row, but does not change otherwise. The function returns *true* if the move operation was successful, otherwise *false*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Examples
See `JScript - Bubble Sort Dynamic Tables`.

Parent

See also

Property: `Parent` as `AuthenticView` (read-only)

Description
Access the view that owns this range object.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Paste

See also

Method: `Paste()` as Boolean

Description
Returns *False* if the copy/paste buffer is empty, or its content cannot replace the current selection.

Otherwise, deletes the current selection, inserts the content of the copy/paste buffer, and returns *True*.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.

PerformAction

See also

Method: `PerformAction` (*eAction* as `SPYAuthenticActions`, *strElementName* as String) as Boolean

Description
`PerformAction` and its related methods, give access to the entry-helper functions of Authentic. This function allows easy and consistent modification of the document content without a need to know exactly where the modification will take place. The beginning of the range object is used to locate the next valid location where the specified action can be performed. If no such location can be found, the method returns *False*. Otherwise, the document gets modified and the range points to the beginning of the modification.

HINT: To find out element names that can be passed as the second parameter use `CanPerformActionWith`.

Errors
2001 The authentic range object, or its related view object is no longer valid.
2005 Invalid address for the return parameter was specified.
2007 Invalid action was specified.

Examples

```
' ------------------------------------------
' Scripting environment - VBScript
' Insert the innermost element
' ------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' we determine the elements that can be inserted at the current position
Dim arrElements()
objRange.CanPerformActionWith spyAuthenticInsertBefore, arrElements

' we insert the first (innermost) element
```

 © 2015 Altova GmbH

```
If UBound(arrElements) >= 0 Then
     objRange.PerformAction spyAuthenticInsertBefore, arrElements(0)
     ' objRange now points to the beginning of the inserted element
     ' we set a default value and position at its end
     objRange.Text = "Hello"
     objRange.ExpandTo(spyAuthenticTag).CollapsToEnd().Select
Else
     MsgBox "Can't insert any elements at current position"
End If
```

Select

See also

Method: `Select()`

Description

Makes this range the current user interface selection. You can achieve the same result using: '*objRange.Parent.Selection = objRange*'

Errors

2001 The authentic range object or its related view object is no longer valid.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' set current selection to end of document
objAuthenticView.DocumentEnd.Select()
```

SelectNext

See also

Method: `SelectNext` (*`eKind`* as `SPYAuthenticElementKind`) as `AuthenticRange`

Description

Selects the element of type `eKind` after the current end of the range. The method returns the modified range object.

Errors

2001 The authentic range object, or its related view object is no longer valid.
2003 Target lies after end of document.
2005 Invalid element kind specified.
Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------------
' Scripting environment - VBScript
' Scan through the whole document word-by-word
```

```
' ------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentBegin
Dim bEndOfDocument
bEndOfDocument = False

On Error Resume Next
While Not bEndOfDocument
     objRange.SelectNext(spyAuthenticWord).Select
     If ((Err.number - vbObjecterror) = 2003) Then
            bEndOfDocument = True
            Err.Clear
     ElseIf (Err.number <> 0) Then
            Err.Raise  ' forward error
     End If
Wend
```

SelectPrevious

See also

Method: `GotoPrevious` (*`eKind`* as `SPYAuthenticElementKind`) as `AuthenticRange`

Description

Selects the element of type `eKind` before the current beginning of the range. The method returns the modified range object.

Errors

2001	The authentic range object, or its related view object is no longer valid.
2004	Target lies before begin of document.
2005	Invalid element kind specified.
	Invalid address for the return parameter was specified.

Examples

```
' ------------------------------------------
' Scripting environment - VBScript
' Scan through the whole document tag-by-tag
' ------------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

Dim objRange
Set objRange = objAuthenticView.DocumentEnd
Dim bEndOfDocument
bBeginOfDocument = False

On Error Resume Next
While Not bBeginOfDocument
```

 © 2015 Altova GmbH

```
      objRange.SelectPrevious(spyAuthenticTag).Select
      If ((Err.number - vbObjecterror) = 2004) Then
            bBeginOfDocument = True
            Err.Clear
      ElseIf (Err.number <> 0) Then
            Err.Raise  ' forward error
      End If
Wend
```

SetElementAttributeValue

See also

Method: SetElementAttributeValue (*strElementName* as String, *strAttributeName* as String, *strAttributeValue* as String)

Description

Set the value of the attribute specified in strAttributeName for the element identified with strElementName. If the attribute is supported but has no value assigned, the empty string is returned. To find out the names of attributes supported by an element, use GetElementAttributeNames, or HasElementAttribute.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid element name was specified. Invalid attribute name was specified. Invalid attribute value was specified.

Examples

```
' ------------------------------------------
' Scripting environment - VBScript
' Get and set element attributes
' ------------------------------------------
Dim objRange
' we assume that the active document is open in authentic view mode
Set objRange = Application.ActiveDocument.AuthenticView.Selection

' first we find out all the elements below the beginning of the range
Dim arrElements
objRange.GetElementHierarchy arrElements

If IsArray(arrElements) Then
      If UBound(arrElements) >= 0 Then
            ' we use the top level element and find out its valid attributes
            Dim arrAttrs()
            objRange.GetElementAttributeNames arrElements(0), arrAttrs

            If UBound(arrAttrs) >= 0 Then
                  ' we retrieve the current value of the first valid attribute
                  Dim strAttrVal
                  strAttrVal = objRange.GetElementAttributeValue
(arrElements(0), arrAttrs(0))
                  msgbox "current value of " & arrElements(0) & "//" &
```

```
  arrAttrs(0) & " is: " & strAttrVal

                    ' we change this value and read it again
                    strAttrVal = "Hello"
                    objRange.SetElementAttributeValue arrElements(0),
  arrAttrs(0), strAttrVal
                    strAttrVal = objRange.GetElementAttributeValue
  (arrElements(0), arrAttrs(0))
                    msgbox "new value of " & arrElements(0) & "//" & arrAttrs(0)
  & " is: " & strAttrVal
              End If
       End If
End If
```

SetFromRange

See also

Method: `SetFromRange` (*`objSrcRange`* as `AuthenticRange`)

Description
Sets the range object to the same beginning and end positions as `objSrcRange`.

Errors

2001	One of the two range objects, is invalid.
2005	Null object was specified as source object.

SetVariableValue

Method: `SetVariableValue(strName as string, strValue as string)`

Return Value
Sets the value (second parameter) of the variable named in the first parameter.

Errors

2201	Invalid object.
2202	No context.
2204	No such variable in scope
2205	Variable cannot be evaluated
2206	Variable returns sequence
2207	Variable read-only
2208	No modification allowed

Text

See also

Property: `Text` as String

© 2015 Altova GmbH

Description
Set or get the textual content selected by the range object.

The number of characters retrieved are not necessarily identical, as there are text cursor positions between the beginning and end of the selected range. Most document elements support an end cursor position different to the beginning cursor position of the following element. Drop-down lists maintain only one cursor position, but can select strings of any length. In the case of radio buttons and check boxes, the text property value holds the string of the corresponding XML element.

If the range selects more then one element, the text is the concatenation of the single texts. XML entities are expanded so that '&' is expected as '&'.

Setting the text to the empty string, does not delete any XML elements. Use `Cut`, `Delete` or `PerformAction` instead.

Errors

2001	The authentic range object or its related view object is no longer valid.
2005	Invalid address for a return parameter was specified.

3.2.6 AuthenticView

See also

Properties	Methods
Application	Goto
AsXMLString	IsRedoEnabled
DocumentBegin	IsUndoEnabled
DocumentEnd	Print
Event	Redo
MarkupVisibility	Undo
Parent	UpdateXMLInstanceEntities
Selection	
XMLDataRoot	
WholeDocument	

Description

AuthenticView and its child objects AuthenticRange and AuthenticDataTransfer provide you with an interface for vista Authentic, which allow easy and consistent modification of document contents. These interfaces replace the following interfaces which are marked now as **obsolete**:

OldAuthenticView (old name was DocEditView)
AuthenticSelection (old name was DocEditSelection, superseded by AuthenticRange)
AuthenticEvent (old name was DocEditEvent)

AuthenticView gives you easy access to specific features such as printing, the multi-level undo buffer, and the current cursor selection, or position.

AuthenticView uses objects of type AuthenticRange to make navigation inside the document straight-forward, and to allow for the flexible selection of logical text elements. Use the properties DocumentBegin, DocumentEnd, or WholeDocument for simple selections, while using the Goto method for more complex selections. To navigate relative to a given document range, see the methods and properties of the AuthenticRange object.

Events

OnBeforeCopy

See also

Event: OnBeforeCopy() as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforeCopy()
     ' On_AuthenticBeforeCopy = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeCopy()
{
     // return false;  /* to disable operation */
```

 © 2015 Altova GmbH

```
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (21, ...)  //  nEventId = 21
```

Description

This event gets triggered before a copy operation gets performed on the document. Return *True* (or nothing) to allow copy operation. Return *False* to disable copying.

OnBeforeCut

See also

Event: `OnBeforeCut()` as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforeCut()
     ' On_AuthenticBeforeCut = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeCut()
{
     // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (20, ...)  //  nEventId = 20
```

Description

This event gets triggered before a cut operation gets performed on the document. Return *True* (or nothing) to allow cut operation. Return *False* to disable operation.

OnBeforeDelete

See also

Event: `OnBeforeDelete()` as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticBeforeDelete()
     ' On_AuthenticBeforeDelete = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticBeforeDelete()
{
     // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (22, ...)  //  nEventId = 22
```

Description
This event gets triggered before a delete operation gets performed on the document. Return *True* (or nothing) to allow delete operation. Return *False* to disable operation.

OnBeforeDrop

See also

Event: `OnBeforeDrop (i_nXPos as Long, i_nYPos as Long, i_ipRange as AuthenticRange, i_ipData as cancelBoolean`

Scripting environment - VBScript:
```
Function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
    ' On_AuthenticBeforeDrop = False  ' to disable operation
End Function
```

Scripting environment - JScript:
```
function On_AuthenticBeforeDrop(nXPos, nYPos, objRange, objData)
{
    // return false;  /* to disable operation */
}
```

IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (11, ...) // nEventId = 11
```

Description
This event gets triggered whenever a previously dragged object gets dropped inside the application window. All event related information gets passed as parameters.

The first two parameters specify the mouse position at the time when the event occurred. The parameter *objRange* passes a range object that selects the XML element below the mouse position. The value of this parameter might be *NULL*. Be sure to check before you access the range object. The parameter *objData* allows to access information about the object being dragged.

Return *False* to cancle the drop operation. Return *True* (or nothing) to continue normal operation.

OnBeforePaste

See also

Event: `OnBeforePaste (objData as Variant, strType as String) as Boolean`

Scripting environment - VBScript:
```
Function On_AuthenticBeforePaste(objData, strType)
    ' On_AuthenticBeforePaste = False  ' to disable operation
End Function
```

Scripting environment - JScript:
```
function On_AuthenticBeforePaste(objData, strType)
{
```


© 2015 Altova GmbH

```
    // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (19, ...)    // nEventId = 19
```

Description

This event gets triggered before a paste operation gets performed on the document. The parameter *strType* is one of "TEXT", "UNICODETEXT" or "IUNKNOWN". In the first two cases *objData* contains a string representation of the object that will be pasted. In the later case, *objData* contains a pointer to an IUnknown COM interface.

Return *True* (or nothing) to allow paste operation. Return *False* to disable operation.

OnBeforeSave

Event: `OnBeforeSave` (SaveAs flag) as Boolean

Description: `OnBeforeSave` gives the opportunity to e.g. warn the user about overwriting the existing XML document, or to make the document read-only when specific circumstances are not met. The event will be fired before the file dialog is shown.

OnDragOver

See also

Event: `OnDragOver` (*nXPos* as Long, *nYPos* as Long, *eMouseEvent* as `SPYMouseEvent`, *objRange* as `AuthenticRange`, *objData* as `AuthenticDataTransfer`) as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange,
objData)
    ' On_AuthenticDragOver = False  ' to disable operation
End Function
```

Scripting environment - JScript:

```
function On_AuthenticDragOver(nXPos, nYPos, eMouseEvent, objRange,
objData)
{
    // return false;  /* to disable operation */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (10, ...)  // nEventId = 10
```

Description

This event gets triggered whenever an object from within or outside of vista Authentic gets dragged with the mouse over the application window. All event related information gets passed as parameters.

The first three parameters specify the mouse position, the mouse button status and the status of

the virtual keys at the time when the event occurred. The parameter *objRange* passes a range object that selects the XML element below the mouse position. The value of this parameter might be *NULL*. Be sure to check before you access the range object. The parameter *objData* allows to access information about the object being dragged.

Return *False* to cancel the drag operation. Return *True* (or nothing) to continue normal operation.

OnKeyboardEvent

See also

Event: `OnKeyboardEvent` (*eKeyEvent* as SPYKeyEvent, *nKeyCode* as Long, *nVirtualKeyStatus* as Long) as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode,
nVirtualKeyStatus)
     ' On_AuthenticKeyboardEvent = True  ' to cancel bubbling of
event
End Function
```

Scripting environment - JScript:

```
function On_AuthenticKeyboardEvent(eKeyEvent, nKeyCode,
nVirtualKeyStatus)
{
     // return true;  /* to cancel bubbling of event */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (30, ...)  //  nEventId = 30
```

Description
This event gets triggered for *WM_KEYDOWN*, *WM_KEYUP* and *WM_CHAR* Windows messages.

The actual message type is available in the *eKeyEvent* parameter. The status of virtual keys is combined in the parameter *nVirtualKeyStatus*. Use the bit-masks defined in the enumeration datatype `SPYVirtualKeyMask`, to test for the different keys or their combinations.

OnLoad

Event: `OnLoad` ()

Description: `OnLoad` can be used e.g. to restrict some AuthenticView functionality, as shown in the example below:

```
function On_AuthenticLoad(  )
{
       // We are disabling all entry helpers in order to prevent user from
manipulating XML tree
       AuthenticView.DisableElementEntryHelper();
       AuthenticView.DisableAttributeEntryHelper();

       // We are also disabling the markup buttons for the same purpose
```

 © 2015 Altova GmbH

```
        AuthenticView.SetToolbarButtonState( 'AuthenticMarkupSmall',
authenticToolbarButtonDisabled );
        AuthenticView.SetToolbarButtonState( 'AuthenticMarkupLarge',
authenticToolbarButtonDisabled );
        AuthenticView.SetToolbarButtonState( 'AuthenticMarkupMixed',
authenticToolbarButtonDisabled );
}
```

In the example the status of the Markup Small, Markup Large, Markup Mixed toolbar buttons are manipulated with the help of button identifiers. See complete list.

OnMouseEvent

See also

Event: OnMouseEvent (*nXPos* as Long, *nYPos* as Long, *eMouseEvent* as SPYMouseEvent, *objRange* as AuthenticRange) as Boolean

Scripting environment - VBScript:

```
Function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
    ' On_AuthenticMouseEvent = True   ' to cancel bubbling of event
End Function
```

Scripting environment - JScript:

```
function On_AuthenticMouseEvent(nXPos, nYPos, eMouseEvent, objRange)
{
    // return true;   /* to cancel bubbling of event */
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (31, ...)  //  nEventId = 31
```

Description

This event gets triggered for every mouse movement and mouse button Windows message.

The actual message type and the mouse buttons status, is available in the *eMouseEvent* parameter. Use the bit-masks defined in the enumeration datatype SPYMouseEvent to test for the different messages, button status, and their combinations.

The parameter *objRange* identifies the part of the document found at the current mouse cursor position. The range objects always selects a complete tag of the document. (This might change in future versions, when a more precise positioning mechanism becomes available). If no selectable part of the document is found at the current position, the range object is *null*.

OnSelectionChanged

See also

Event: OnSelectionChanged (*objNewSelection* as AuthenticRange)

Scripting environment - VBScript:

```
Function On_AuthenticSelectionChanged (objNewSelection)
End Function
```

Scripting environment - JScript:

```
function On_AuthenticSelectionChanged (objNewSelection)
{
}
```

IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (23, ...)  // nEventId = 23
```

Description
This event gets triggered whenever the selection in the user interface changes.

OnToolbarButtonClicked

Event: `OnToolbarButtonClicked` (Button identifier)

Description: `OnToolbarButtonClicked` is fired when a toolbar button was clicked by user. The parameter button identifier helps to determine which button was clicked. The list of predefined button identifiers is below:

- `AuthenticPrint`
- `AuthenticPrintPreview`
- `AuthenticUndo`
- `AuthenticRedo`
- `AuthenticCut`
- `AuthenticCopy`
- `AuthenticPaste`
- `AuthenticClear`
- `AuthenticMarkupHide`
- `AuthenticMarkupLarge`
- `AuthenticMarkupMixed`
- `AuthenticMarkupSmall`
- `AuthenticValidate`
- `AuthenticChangeWorkingDBXMLCell`
- `AuthenticSave`
- `AuthenticSaveAs`
- `AuthenticReload`
- `AuthenticTableInsertRow`
- `AuthenticTableAppendRow`
- `AuthenticTableDeleteRow`
- `AuthenticTableInsertCol`
- `AuthenticTableAppendCol`
- `AuthenticTableDeleteCol`
- `AuthenticTableJoinCellRight`
- `AuthenticTableJoinCellLeft`
- `AuthenticTableJoinCellAbove`
- `AuthenticTableJoinCellBelow`
- `AuthenticTableSplitCellHorizontally`
- `AuthenticTableSplitCellVertically`
- `AuthenticTableAlignCellContentTop`
- `AuthenticTableCenterCellVertically`

© 2015 Altova GmbH

- AuthenticTableAlignCellContentBottom
- AuthenticTableAlignCellContentLeft
- AuthenticTableCenterCellContent
- AuthenticTableAlignCellContentRight
- AuthenticTableJustifyCellContent
- AuthenticTableInsertTable
- AuthenticTableDeleteTable
- AuthenticTableProperties
- AuthenticAppendRow
- AuthenticInsertRow
- AuthenticDuplicateRow
- AuthenticMoveRowUp
- AuthenticMoveRowDown
- AuthenticDeleteRow
- AuthenticDefineEntities
- AuthenticXMLSignature

For custom buttons the user might add his own identifiers. Please, note that the user must take care, as the identifiers are not checked for uniqueness. The same identifiers can be used to identify buttons in the `Set/GetToolbarState()` COM API calls. By adding code for different buttons, the user is in the position to completely redefine the AuthenticView toolbar behavior, adding own methods for table manipulation, etc.

OnToolbarButtonExecuted

Event: `OnToolbarButtonExecuted` (Button identifier)

Description: `OnToolbarButtonClicked` is fired when a toolbar button was clicked by user. The parameter button identifier helps to determine which button was clicked. See the list of predefined button identifiers.

`OnToolbarButtonExecuted` is fired after the toolbar action was executed. It is useful e.g. to add update code, as shown in the example below:

```
//event fired when a toolbar button action was executed
function On_AuthenticToolbarButtonExecuted( varBtnIdentifier )
{
	// After whatever command user has executed - make sure to update toolbar
button states
	UpdateOwnToolbarButtonStates();
}
```

In this case `UpdateOwnToolbarButtonStates` is a user function defined in the Global Declarations.

OnUserAddedXMLNode

Event: `OnUserAddedXMLNode` (XML node)

Description: `OnUserAddedXMLNode` will be fired when the user adds an XML node as a primary action. This happens in the situations, where the user clicks on

- auto-add hyperlinks (see example OnUserAddedXMLNode.sps)

- the Insert…, Insert After…, Insert Before… context menu items
- Append row, Insert row toolbar buttons
- Insert After…, Insert Before… actions in element entry helper (outside StyleVision)

The event doesn't get fired on Duplicate row, or when the node was added externally (e.g. via COM API), or on Apply (e.g. Text State Icons), or when in XML table operations or in DB operations.

The event parameter is the XML node object, which was added giving the user an opportunity to manipulate the XML node added. An elaborate example for an event handler can be found in the `OnUserAddedXMLNode.sps` file.

Application

See also

Property: `Application` as `Application` (read-only)

Description
Accesses the Authentic Desktop application object.

Errors

2000	The authentic view object is no longer valid.
2005	Invalid address for the return parameter was specified.

AsXMLString

See also

Property: `AsXMLString` as String

Description
Returns or sets the document content as an XML string. Setting the content to a new value does not change the schema file or sps file in use. If the new `XMLString` does not match the actual schema file error 2011 gets returned.

Errors

2000	The authentic view object is no longer valid.
2011	`AsXMLString` was set to a value which is no valid XML for the current schema file.

ContextMenu

Property: `ContextMenu()` as `ContextMenu`

Description
The property `ContextMenu` gives access to customize the context menu. The best place to do it is in the event handler `OnContextMenuActivated`.

Errors

2000	Invalid object.
2005	Invalid parameter.

© 2015 Altova GmbH

CreateXMLNode

Method: `CreateXMLNode` (*`nKind`* as `SPYXMLDataKind`) as `XMLData`

Return Value
The method returns the new `XMLData` object.

Description
To create a new `XMLData` object use the `CreateXMLNode()` method.

Errors
2000 Invalid object.
2012 Cannot create XML node.

DisableAttributeEntryHelper

Method: `DisableAttributeEntryHelper()`

Description
`DisableAttributeEntryHelper()` disables the attribute entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors
2000 Invalid object.

DisableElementEntryHelper

Method: `DisableElementEntryHelper()`

Description
`DisableElementEntryHelper()` disables the element entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors
2000 Invalid object.

DisableEntityEntryHelper

Method: `DisableEntityEntryHelper()`

Description
`DisableEntityEntryHelper()` disables the entity entry helper in XMLSpy, Authentic Desktop and Authentic Browser plug-in.

Errors
2000 Invalid object.

DocumentBegin

See also

Property: `DocumentBegin` as `AuthenticRange` (read-only)

Description
Retrieve a range object that points to the beginning of the document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

DocumentEnd

See also

Property: `DocumentEnd` as `AuthenticRange` (read-only)

Description
Retrieve a range object that points to the end of the document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

DoNotPerformStandardAction

Method: `DoNotPerformStandardAction ()`

Description
`DoNotPerformStandardAction()` serves as cancel bubble for macros, and stops further execution after macro has finished.

Errors
2000 Invalid object.

EvaluateXPath

Method: `EvaluateXPath` (XMLData as XMLData, strExpression as string) strValue as string

Return Value
The method returns a string

Description
`EvaluateXPath()` executes an XPath expressions with the given XML context node. The result is returned as string, in the case of a sequence it is a space-separated string.

Errors
2000 Invalid object.
2005 Invalid parameter.
2008 Internal error.

© 2015 Altova GmbH

2013 XPath error.

Event

See also

Property: `Event` as `AuthenticEvent` (read-only)

Description
This property gives access to parameters of the last event in the same way as `OldAuthenticView.event` does. Since all events for the scripting environment and external clients are now available with parameters this `Event` property should only be used from within IDE-Plugins.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

EventContext

Property: `EventContext()` as `EventContext`

Description
`EventContext` property gives access to the running macros context. See the `EventContext` interface description for more details.

Errors
2000 Invalid object.

GetToolbarButtonState

Method: `GetToolbarButtonState` (`ButtonIdentifier` as `string`) as `AuthenticToolbarButtonState`

Return Value
The method returns `AuthenticToolbarButtonState`

Description
`Get/SetToolbarButtonState` queries the status of a toolbar button, and lets the user disable or enable the button, identified via its button identifier (see list above). One usage is to disable toolbar buttons permanently. Another usage is to put `SetToolbarButtonState` in the `OnSelectionChanged` event handler, as toolbar buttons are updated regularly when the selection changes in the document.

Toolbar button states are given by the listed enumerations.

The default state means that the enable/disable of the button is governed by AuthenticView. When the user sets the button state to enable or disable, the button remains in that state as long as the user does not change it.

Errors

2000 Invalid object.
2005 Invalid parameter.
2008 Internal error.
2014 Invalid button identifier.

Goto

See also

Method: Goto (*eKind* as SPYAuthenticElementKind, *nCount* as Long, *eFrom* as *SPYAuthenticDocumentPosition*) as *AuthenticRange*

Description
Retrieve a range object that points to the beginning of the *nCount* element of type *eKind*. The start position is defined by the parameter *eFrom*. Use positive values for *nCount* to navigate to the document end. Use negative values to navigate towards the beginning of the document.

Errors
2000 The authentic view object is no longer valid.
2003 Target lies after end of document.
2004 Target lies before beginning of document.
2005 Invalid element kind specified.
The document position to start from is not one of *spyAuthenticDocumentBegin* or *spyAuthenticDocumentEnd*.
Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

On Error Resume Next
Dim objRange
' goto beginning of first table in document
Set objRange = objAuthenticView.Goto (spyAuthenticTable, 1,
spyAuthenticDocumentBegin)
If (Err.number = 0) Then
     objRange.Select()
Else
     MsgBox "No table found in document"
End If
```

IsRedoEnabled

See also

Property: IsRedoEnabled as Boolean (read-only)

Description
True if redo steps are available and Redo is possible.

 © 2015 Altova GmbH

Errors

2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

IsUndoEnabled

See also

Property: `IsUndoEnabled` as Boolean (read-only)

Description
True if undo steps are available and `Undo` is possible.

Errors

2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

MarkupVisibility

See also

Property: `MarkupVisibility` as `SPYAuthenticMarkupVisibility`

Description
Set or get current visibility of markup.

Errors

2000 The authentic view object is no longer valid.
2005 Invalid enumeration value was specified.
Invalid address for the return parameter was specified.

Parent

See also

Property: `Parent` as `Document` (read-only)

Description
Access the document shown in this view.

Errors

2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Print

See also

Method: `Print` (*bWithPreview* as Boolean, *bPromptUser* as Boolean)

Description
Print the document shown in this view. If *bWithPreview* is set to *True*, the print preview dialog pops up. If *bPromptUser* is set to *True*, the print dialog pops up. If both parameters are set to *False*, the document gets printed without further user interaction.

Errors
2000 The authentic view object is no longer valid.

Redo

See also

Method: `Redo()` as Boolean

Description
Redo the modification undone by the last undo command.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

Selection

See also

Property: `Selection` as `AuthenticRange`

Description
Set or get current text selection in user interface.

Errors
2000 The authentic view object is no longer valid.
2002 No cursor selection is active.
2005 Invalid address for the return parameter was specified.

Examples

```
' ---------------------------------------
' Scripting environment - VBScript
' ---------------------------------------
Dim objAuthenticView
' we assume that the active document is open in authentic view mode
Set objAuthenticView = Application.ActiveDocument.AuthenticView

' if we are the end of the document, re-start at the beginning
If (objAuthenticView.Selection.IsEqual(objAuthenticView.DocumentEnd)) Then
     objAuthenticView.Selection = objAuthenticView.DocumentBegin
Else
     ' objAuthenticView.Selection =
objAuthenticView.Selection.GotoNextCursorPosition()
     ' or shorter:
      objAuthenticView.Selection.GotoNextCursorPosition().Select
End If
```

 © 2015 Altova GmbH

SetToolbarButtonState

Method: `SetToolbarButtonState` (`ButtonIdentifier` as string, `AuthenticToolbarButtonState` state)

Description

`Get/SetToolbarButtonState` queries the status of a toolbar button, and lets the user disable or enable the button, identified via its button identifier (see list above). One usage is to disable toolbar buttons permanently. Another usage is to put `SetToolbarButtonState` in the `OnSelectionChanged` event handler, as toolbar buttons are updated regularly when the selection changes in the document.

Toolbar button states are given by the listed enumerations.

The default state means that the enable/disable of the button is governed by AuthenticView. When the user sets the button state to enable or disable, the button remains in that state as long as the user does not change it.

Errors

2000	Invalid object.
2008	Internal error.
2014	Invalid button identifier.

Undo

See also

Method: Undo() as Boolean

Description

Undo the last modification of the document from within this view.

Errors

2000	The authentic view object is no longer valid.
2005	Invalid address for the return parameter was specified.

UpdateXMLInstanceEntities

See also

Method: `UpdateXMLInstanceEntities()`

Description

Updates the internal representation of the declared entities, and refills the entry helper. In addition, the validator is reloaded, allowing the XML file to validate correctly. Please note that this may also cause schema files to be reloaded.

Errors

The method never returns an error.

Example

```
// ----------------------------------------
// Scripting environment - JavaScript
// ----------------------------------------
if(Application.ActiveDocument && (Application.ActiveDocument.CurrentViewMode
== 4))
{
     var objDocType;
     objDocType =
Application.ActiveDocument.DocEditView.XMLRoot.GetFirstChild(10);

     if(objDocType)
     {
            var objEntity = Application.ActiveDocument.CreateChild(14);
            objEntity.Name = "child";
            objEntity.TextValue = "SYSTEM \"child.xml\"";
            objDocType.AppendChild(objEntity);


Application.ActiveDocument.AuthenticView.UpdateXMLInstanceEntities();
     }
}
```

WholeDocument

See also

Property: `WholeDocument` as `AuthenticRange` (read-only)

Description
Retrieve a range object that selects the whole document.

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

XMLDataRoot

See also

Property: `XMLDataRoot` as `XMLData` (read-only)

Description
Returns or sets the top-level XMLData element of the current document. This element typically describes the document structure and would be of kind spyXMLDataXMLDocStruct, spyXMLDataXMLEntityDocStruct or spyXMLDataDTDDocStruct..

Errors
2000 The authentic view object is no longer valid.
2005 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

3.2.7 CodeGeneratorDlg

See also
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Properties and Methods

Standard automation properties
Application
Parent

Programming language selection properties
ProgrammingLanguage
TemplateFileName

Settings for C++ code
CPPSettings_DOMType
CPPSettings_LibraryType
CPPSettings_UseMFC
CPPSettings_GenerateVC6ProjectFile
CPPSettings_GenerateVSProjectFile

Settings for C# code
CSharpSettings_ProjectType

Dialog handling for above code generation properties
PropertySheetDialogAction

Output path selection properties
OutputPath
OutputPathDialogAction

Presentation of result
OutputResultDialogAction

Description
Use this object to configure the generation of program code for schema files. The method GenerateProgramCode expects a CodeGeneratorDlg as parameter to configure code generation as well as the associated user interactions.

Application

See also
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

CPPSettings_DOMType

Property: `CPPSettings_DOMType` as `SPYDOMType`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

CPPSettings_GenerateVC6ProjectFile

Property: `CPPSettings_GenerateVC6ProjectFile` as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

CPPSettings_GenerateGCCMakefile

Property: `CPPSettings_GenerateGCCMakefile` as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Creates makefiles to compile the generated code under Linux with GCC.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

© 2015 Altova GmbH

CPPSettings_GenerateVSProjectFile

Property: `CSharpSettings_GenerateVSProjectFile` as `SPYProjectType`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code. Only `spyVisualStudio2005Project (=4)` and `spyVisualStudio2008Project (=5)` and `spyVisualStudio2010Project (=6)` are valid project types.

Errors

2200	The object is no longer valid.
2201	Invalid action passed as parameter or an invalid address was specified for the return parameter.

CPPSettings_LibraryType

Property: `CPPSettings_LibraryType` as `SPYLibType`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors

2200	The object is no longer valid.
2201	Invalid action passed as parameter or an invalid address was specified for the return parameter.

CPPSettings_UseMFC

Property: `CPPSettings_UseMFC` as Boolean
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines one of the settings that configure generation of C++ code.

Errors

2200	The object is no longer valid.
2201	Invalid action passed as parameter or an invalid address was specified for the return parameter.

CSharpSettings_ProjectType

Property: `CSharpSettings_ProjectType` as `SPYProjectType`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description

Defines the only setting to configure generation of C# code.

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

OutputPath

Property: `OutputPath` as String
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Selects the base directory for all generated code.

Errors

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

OutputPathDialogAction

Property: `OutputPathDialogAction` as `SPYDialogAction`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog for selecting the code generation output path gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current value of the OutputPath property as default. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

OutputResultDialogAction

Property: `OutputResultDialogAction` as `SPYDialogAction`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog that asks to show the result of the code generation process gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors

2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

© 2015 Altova GmbH

Parent

See also
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

ProgrammingLanguage

Property: `ProgrammingLanguage` as `ProgrammingLanguage`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Selects the output language for the code to be generated.

CAUTION: Setting this property to one of C++, C# or Java, changes the property TemplateFileName to the appropriate template file delivered with Authentic Desktop as well. If you want to generate C++, C# or Java code based on your own templates, set first the programming language and then select your template file.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

PropertySheetDialogAction

Property: `PropertySheetDialogAction` as `SPYDialogAction`
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Defines how the sub-dialog that configures the code generation process gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current values as defaults. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
2200 The object is no longer valid.
2201 Invalid action passed as parameter or an invalid address was specified for the return parameter.

TemplateFileName

Property: `TemplateFileName` as String
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Selects the code generation template file. Authentic Desktop comes with template files for C++, C# or Java in the SPL folder of your installation directory.

Setting this property to one of the code generation template files of your Authentic Desktop installation automatically sets the ProgrammingLanguage property to its appropriate value.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

3.2.8 DatabaseConnection

See also

Properties for import and export
File or
ADOConnection or
ODBCConnection

Properties for import only
DatabaseKind
SQLSelect
AsAttributes
ExcludeKeys
IncludeEmptyElements
NumberDateTimeFormat
NullReplacement
CommentIncluded

Properties for export only
CreateMissingTables
CreateNew
TextFieldLen
DatabaseSchema

Properties for XML Schema from DB Structure generation
PrimaryKeys
ForeignKeys
UniqueKeys
SchemaExtensionType
SchemaFormat
ImportColumnsType

Description
DatabaseConnection specifies the parameters for the database connection.

Please note that the properties of the DatabaseConnection interface are referring to the settings of the import and export dialogs of Authentic Desktop.

ADOConnection

See also

Property: ADOConnection as String

Description
The property ADOConnection contains a connection string. Either use this property or ODBCConnection or File to refer to a database.

Errors
No error codes are returned.

Example

```
Dim objSpyConn As DatabaseConnection
Set objSpyConn = objSpy.GetDatabaseSettings

Dim objADO As DataLinks
Set objADO = CreateObject("DataLinks")

If Not (objADO Is Nothing) Then
 Dim objConn As Connection
 Set objConn = objADO.PromptNew
 objSpyConn.ADOConnection = objConn.ConnectionString
End If
```

AsAttributes

See also

Property: `AsAttributes` as `Boolean`

Description
Set `AsAttributes` to true if you want to initialize all import fields to be imported as attributes. Default is false and will initialize all fields to be imported as elements. This property is used only in calls to `Application.GetDatabaseImportElementList`.

Errors
No error codes are returned.

CommentIncluded

See also

Property: CommentIncluded as Boolean

Description
This property tells whether additional comments are added to the generated XML. Default is true. This property is used only when importing from databases.

Errors
No error codes are returned.

CreateMissingTables

See also

Property: `CreateMissingTables` as `Boolean`

Description
If `CreateMissingTables` is true, tables which are not already defined in the export database will be created during export. Default is true. This property is used only when exporting to databases.

Errors
No error codes are returned.

© 2015 Altova GmbH

CreateNew

See also

Property: `CreateNew` as `Boolean`

Description
Set `CreateNew` true if you want to create a new database on export. Any existing database will be overwritten. See also `DatabaseConnection.File`. Default is false. This property is used only when exporting to databases.

Errors
No error codes are returned.

DatabaseKind

See also

Property: `DatabaseKind` as `SPYDatabaseKind`

Description
Select the kind of database that gets access. The default value is `spyDB_Unspecified(7)` and is sufficient in most cases. This property is used only when importing from databases.

Errors
No error codes are returned.

DatabaseSchema

See also

Property: DatabaseSchema as String

Description
This property specifies the Schema used for export in Schema aware databases. Default is "". This property is used only when exporting to databases.

Errors
No error codes are returned.

ExcludeKeys

See also

Property: `ExcludeKeys` as `Boolean`

Description
Set `ExcludeKeys` to true if you want to exclude all key columns from the import data. Default is false. This property is used only when importing from databases.

Errors

No error codes are returned.

File

See also

Property: `File` as `String`

Description
The property `File` sets the path for the database during export or import. This property can only be used in conjunction with a Microsoft Access database. Either use this property or `ODBCConnection` or `ADOConnection` to refer to the database.

Errors
No error codes are returned.

ForeignKeys

See also

Property: `ForeignKeys` as `Boolean`

Description
Specifies whether the Foreign Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

ImportColumnsType

See also

Property: `ImportColumnsType` as SPYImportColumnsType

Description
Defines if column information from the DB is saved as element or attribute in the XML Schema. Default is as element. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

IncludeEmptyElements

See also

Property: `IncludeEmptyElements` as `Boolean`

Description
Set `IncludeEmptyElements` to false if you want to exclude all empty elements. Default is true. This property is used only when importing from databases.

© 2015 Altova GmbH

Errors
No error codes are returned.

NullReplacement

See also

Property: NullReplacement as String

Description
This property contains the text value that is used during import for empty elements (null values). Default is "". This property is used only when importing from databases.

Errors
No error codes are returned.

NumberDateTimeFormat

See also

Property: `NumberDateTimeFormat` as `SPYNumberDateTimeFormat`

Description
The property `NumberDateTimeFormat` sets the format of numbers and date- and time-values. Default is `spySystemLocale`. This property is used only when importing from databases.

Errors
No error codes are returned.

ODBCConnection

See also

Property: `ODBCConnection` as `String`

Description
The property `ODBCConnection` contains a ODBC connection string. Either use this property or `ADOConnection` or `File` to refer to a database.

Errors
No error codes are returned.

PrimaryKeys

See also

Property: `PrimaryKeys` as `Boolean`

Description
Specifies whether the Primary Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

SchemaExtensionType

See also

Property: `SchemaExtensionType` as SPYSchemaExtensionType

Description
Defines the Schema extension type used during the Schema generation. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

SchemaFormat

See also

Property: `SchemaFormat` as SPYSchemaFormat

Description
Defines the Schema format used during the Schema generation. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

SQLSelect

See also

Property: `SQLSelect` as `String`

Description
The SQL query for the import is stored in the property `SQLSelect`. This property is used only when importing from databases.

Errors
No error codes are returned.

TextFieldLen

See also

Property: `TextFieldLen` as `long`

Description
The property `TextFieldLen` sets the length for created text fields during the export. Default is 255. This property is used only when exporting to databases.

© 2015 Altova GmbH

Errors
No error codes are returned.

UniqueKeys

See also

Property: `UniqueKeys` as `Boolean`

Description
Specifies whether the Unique Keys constraint is created or not. Default is true. This property is used only when creating a XML Schema from a DB structure.

Errors
No error codes are returned.

3.2.9 Dialogs

See also

Properties and Methods

Standard automation properties
Application
Parent

Various dialog objects
CodeGeneratorDlg
FileSelectionDlg
SchemaDocumentationDlg
GenerateSampleXMLDlg
DTDSchemaGeneratorDlg
FindInFilesDlg

Description
The Dialogs object provides access to different built-in dialogs of Authentic Desktop. These dialog objects allow to initialize the fields of user dialogs before they get presented to the user or allow to simulate complete user input by your program.

Application

See also

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2300 The object is no longer valid.
2301 Invalid address for the return parameter was specified.

CodeGeneratorDlg

See also
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Property: CodeGeneratorDlg as CodeGeneratorDlg (read-only)

Description
Get a new instance of a code generation dialog object. You will need this object to pass the necessary parameters to the code generation methods. Initial values are taken from last usage of the code generation dialog.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

FileSelectionDlg

See also

Property: `FileSelectionDlg` as `FileSelectionDlg` (read-only)

Description
Get a new instance of a file selection dialog object.

File selection dialog objects are passed to you with the some events that signal opening or saving of documents and projects.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

Parent

See also

Property: `Parent` as `Application` (read-only)

Description
Access the Authentic Desktop application object.

Errors
2300 The object is no longer valid.
2301 Invalid address for the return parameter was specified.

SchemaDocumentationDlg

See also

Property: `SchemaDocumentationDlg` as `SchemaDocumentationDlg` (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of schema documentation. See Document.GenerateSchemaDocumentation for its usage.

Errors
2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

GenerateSampleXMLDlg

See also

Property: `GenerateSampleXMLDlg` as `GenerateSampleXMLDlg` (read-only)

Description
Get a new instance of a dialog object that parameterizes generation of a sample XML based on a

W3C schema or DTD. See GenerateSampleXML for its usage.

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

DTDSchemaGeneratorDlg

See also

Property: `DTDSchemaGeneratorDlg` as `DTDSchemaGeneratorDlg` (read-only)

Description

Get a new instance of a dialog object that parameterizes generation of a schema or DTD. See Document.GenerateDTDOrSchemaEx for its usage.

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

FindInFilesDlg

See also

Property: FindInFilesDlg as FindInFilesDlg (read-only)

Description

Get a new instance of a dialog object that parameterizes the search (or replacement) of strings in files. See Application.FindInFiles for its usage.

Errors

2300 The Dialogs object or one of its parents is no longer valid.
2301 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

3.2.10 Document

See also

Properties and Methods

Standard automation properties
Application
Parent

Various document properties and methods
SetActiveDocument
Encoding
SetEncoding (obsolete)
Suggestions

XML validation
IsValid
SetExternalIsValid

Document conversion and transformation
AssignDTD
AssignSchema
AssignXSL
AssignXSLFO
ConvertDTDOrSchema
ConvertDTDOrSchemaEx
GenerateDTDOrSchema
GenerateDTDOrSchemaEx
CreateSchemaDiagram
ExecuteXQuery
TransformXSL
TransformXSLEx
TransformXSLFO
GenerateProgramCode (Enterprise Edition only)
GenerateSchemaDocumentation
GenerateSampleXML
ConvertToWSDL20

Document export
GetExportElementList
ExportToText
ExportToDatabase
CreateDBStructureFromXMLSchema
GetDBStructureList

File saving and naming
FullName
Name
Path
GetPathName (obsolete)
SetPathName (obsolete)
Title

IsModified
Saved
SaveAs
Save
SaveInString
SaveToURL
Close

View access
CurrentViewMode
SwitchViewMode
AuthenticView
GridView
DocEditView (obsolete)

Access to XMLData
RootElement
DataRoot
CreateChild
UpdateViews
StartChanges
EndChanges
UpdateXMLData

Description
Document objects represent XML documents opened in Authentic Desktop.

Use one of the following properties to access documents that are already open Authentic Desktop:
Application.ActiveDocument
Application.Documents

Use one of the following methods to open a new document in Authentic Desktop:
Documents.OpenFile
Documents.OpenURL
Documents.OpenURLDialog
Documents.NewFile
Documents.NewFileFromText
SpyProjectItem.Open
Application.ImportFromDatabase
Application.ImportFromSchema
Application.ImportFromText
Application.ImportFromWord
Document.ConvertDTDOrSchema
Document.GenerateDTDOrSchema

Events

OnBeforeSaveDocument

See also

© 2015 Altova GmbH

Event: OnBeforeSaveDocument(*objDocument* as Document, *objDialog* as FileSelectionDlg)

XMLSpy scripting environment - VBScript:

```
Function On_BeforeSaveDocument(objDocument, objDialog)
End Function

' old handler - now obsolete
' return string to save to new file name
' return empty string to cancel save operation
' return nothing to save to original name
Function On_SaveDocument(objDocument, strFilePath)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeSaveDocument(objDocument, objDialog)
{
}

// old handler - now obsolete
// return string to save to new file name
// return empty string to cancel save operation
// return nothing to save to original name
function On_SaveDocument(objDocument, strFilePath)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (27, ...)  // nEventId = 27
```

Description

This event gets fired on any attempt to save a document. The file selection dialog object is initialized with the name chosen for the document file. You can modify this selection. To continue saving the document leave the FileSelectionDlg.DialogAction property of *io_objDialog* at its default value spyDialogOK. To abort saving of the document set this property to spyDialogCancel.

OnBeforeCloseDocument

See also

Event: OnBeforeCloseDocument(*objDocument* as Document)as Boolean

XMLSpy scripting environment - VBScript:

```
Function On_BeforeCloseDocument(objDocument)
     ' On_BeforeCloseDocument = False  ' to prohibit closing of
document
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeCloseDocument(objDocument)
{
```

```
    // return false;  /* to prohibit closing of document */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (28, ...)  //  nEventId = 28
```

Description

This event gets fired on any attempt to close a document. To prevent the document from being closed return false.

OnBeforeValidate

See also

Event: `OnBeforeValidate(objDocument as Document, bOnLoading as Boolean, bOnCommand as Boolean) as Boolean`

XMLSpy scripting environment - VBScript:

```
Function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
    On_BeforeValidate = bCancelDefaultValidation    'set by the
script if necessary
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeValidate(objDocument, bOnLoading, bOnCommand)
{
    return bCancelDefaultValidation     //set by the script if
necessary
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (32, ...)  //  nEventId = 32
```

Description

This event gets fired before the document is validated. It is possible to suppress the default validation by returning false from the event handler. In this case the script should also set the validation result using the SetExternalIsValid method.

`bOnLoading` is true if the the event is raised on the initial validation on loading the document.

`bOnCommand` is true whenever the user selected the Validate command from the Toolbar or menu.

Available with TypeLibrary version 1.5

OnCloseDocument

See also

Event: `OnCloseDocument(objDocument as Document)`

© 2015 Altova GmbH

XMLSpy scripting environment - VBScript:

```
Function On_Close Document(objDocument)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_Close Document(objDocument)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (8, ...)   // nEventId = 8
```

Description

This event gets fired as a result of closing a document. Do not modify the document from within this event.

OnViewActivation

See also

Event: `OnViewActivation(objDocument` as `Document`, *`eViewMode`* as `SPYViewModes`, *`bActivated`* as Boolean)

XMLSpy scripting environment - VBScript:

```
Function On_ViewActivation(objDocument, eViewMode, bActivated)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_ViewActivation(objDocument, eViewMode, bActivated)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (29, ...)  // nEventId = 29
```

Description

This event gets fired whenever a view of a document becomes visible (i.e. becomes the active view) or invisible (i.e. another view becomes the active view or the document gets closed). However, the first view activation event after a document gets opened cannot be received, since there is no document object to get the event from. Use the `Application.OnDocumentOpened` event instead.

Application

See also

Property: `Application` as `Application` (read-only)

Description

Accesses the Authentic Desktop application object.

Errors

1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

AssignDTD

See also

Method: `AssignDTD(strDTDFile as String, bDialog as Boolean)`

Description

The method places a reference to the DTD file "strDTDFile" into the document. Note that no error occurs if the file does not exist, or is not accessible. If `bDialog` is true Authentic Desktop presents a dialog to set the file.

Errors

1400 The object is no longer valid.
1409 You are not allowed to assign a DTD to the document.

AssignSchema

See also

Method: `AssignSchema (strSchemaFile as String, bDialog as Boolean)`

Description

The method places a reference to the schema file "strSchemaFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If `bDialog` is true Authentic Desktop presents a dialog to set the file.

Errors

1400 The object is no longer valid.
1409 You are not allowed to assign a schema file to the document.

AssignXSL

See also

Method: `AssignXSL (strXSLFile as String, bDialog as Boolean)`

Description

The method places a reference to the XSL file "strXSLFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If `bDialog` is true Authentic Desktop presents a dialog to set the file.

Errors

1400 The object is no longer valid.
1409 You are not allowed to assign an XSL file to the document.

© 2015 Altova GmbH

AssignXSLFO

See also

Method: AssignXSLFO (*strXSLFOFile* as String, *bDialog* as Boolean)

Description
The method places a reference to the XSLFO file "strXSLFile" into the document. Note that no error occurs if the file does not exist or is not accessible. If bDialog is true Authentic Desktop presents a dialog to set the file.

Errors
1400 The object is no longer valid.
1409 You are not allowed to assign an XSL file to the document.

AsXMLString

See also

Property: AsXMLString as String

Description
This property can be used to get or set the document content.

Errors
1400 The document object is no longer valid.
1404 Cannot create XMLData object.
1407 View mode cannot be switched.

AuthenticView

See also

Method: AuthenticView as AuthenticView (read-only)

Description
Returns an object that gives access to properties and methods specific to Authentic view. The object returned is only valid if the current document is opened in Authentic view mode. The lifetime of an object ends with the next view switch. Any attempt to access objects or any of its children afterwards will result in an error indicating that the object is invalid.

Errors
1400 The object is no longer valid.
1417 Document needs to be open in authentic view mode.

Examples

```
' ---------------------------------------
' XMLSpy scripting environment - VBScript
' secure access to authentic view object
' ---------------------------------------
Dim objDocument
Set objDocument = Application.ActiveDocument
If (Not objDocument Is Nothing) Then
```

```
	' we have an active document, now check for view mode
	If (objDocument.CurrentViewMode <> spyViewAuthentic) Then
		If (Not objDocument.SwitchViewMode (spyViewAuthentic)) Then
			MsgBox "Active document does not support authentic view
mode"
		Else
			' now it is safe to access the authentic view object
			Dim objAuthenticView
			Set objAuthenticView = objDocument.AuthenticView
			' now use the authentic view object

		End If
	End If
Else
	MsgBox "No document is open"
End If
```

Close

See also

Method: `Close` (*`bDiscardChanges`* as Boolean)

Description

To close the document call this method. If `bDiscardChanges` is true and the document is modified, the document will be closed but not saved.

Errors

1400 The object is no longer valid.
1401 Document needs to be saved first.

ConvertDTDOrSchema

See also

Method: `ConvertDTDOrSchema` (*`nFormat`* as `SPYDTDSchemaFormat`, *`nFrequentElements`* as `SPYFrequentElements`)

Parameters

`nFormat`
Sets the schema output format to DTD or W3C.

`nFrequentElements`
Create complex elements as elements or complex types.

Description

`ConvertDTDOrSchema` takes an existing schema format and converts it into a different format. For a finer tuning of DTD/XSD conversion, use ConvertDTDOrSchemaEx.

Errors

© 2015 Altova GmbH

1400 The object is no longer valid.
1412 Error during conversion. In the case of DTD to DTD or XSD to XSD conversion, the following errors are returned: *DTD to DTD conversion is not supported. Please use function FlattenDTDOrSchema instead* and *Schema to schema conversion is not supported. Please use function FlattenDTDOrSchema instead*.

ConvertDTDOrSchemaEx

See also

Method: `ConvertDTDOrSchemaEx` (*`nFormat`* as `SPYDTDSchemaFormat`, *`nFrequentElements`* as `SPYFrequentElements`, `sOutputPath` as `String`, `nOutputPathDialogAction` as `SPYDialogAction`)

Parameters

`nFormat`
Sets the schema output format to DTD, or W3C.

`nFrequentElements`
Create complex elements as elements or complex types.

`sOutputPath`
The file path for the newly generated file.

`nOutputPathDialogAction`
Defines the dialog interaction for this call.

Description

`ConvertDTDOrSchemaEx` takes an existing schema format and converts it into a different format.

Errors

1400 The object is no longer valid.
1412 Error during conversion. In the case of DTD to DTD or XSD to XSD conversion, the following errors are returned: *DTD to DTD conversion is not supported. Please use function FlattenDTDOrSchema instead* and *Schema to schema conversion is not supported. Please use function FlattenDTDOrSchema instead*.

ConvertToWSDL20

Method: `ConvertToWSDL20` (*`sFilePath`* as String, *bShowDialogs* as Boolean)

Parameters

`sFilePath`
This specifies the file name of the converted WSDL. In case the source WSDL includes files which also must be converted, then only the directory part of the given path is used and the file names are generated automatically.

`bShowDialogs`
Defines whether file/folder selection dialogs are shown.

Description
Converts the WSDL 1.1 document to a WSDL 2.0 file. It will also convert any referenced WSDL files that are referenced from within this document. Note that this functionality is limited to WSDL View only. See `Document.CurrentViewMode`. and `SPYViewModes`.

Errors
1400 The document object is no longer valid.
1407 Invalid parameters have been passed or an empty file name has been specified as output target.
1417 The document is not opened in WSDL view, maybe it is not an '.wsdl' file.
1421 Feature is not available in this edition.
1433 WSDL 1.1 to WSDL 2.0 conversion failed.

CreateChild

See also

Method: `CreateChild` (*`nKind`* as `SPYXMLDataKind`) as `XMLData`

Return Value
The method returns the new `XMLData` object.

Description
To create a new `XMLData` object use the `CreateChild()` method.

Errors
1400 The object is no longer valid.
1404 Cannot create XMLData object.
1407 Invalid address for the return parameter was specified.

CreateDBStructureFromXMLSchema

See also

Method: `CreateDBStructureFromXMLSchema` (*`pDatabase`* as `DatabaseConnection`, *`pTables`* as `ElementList`, *`bDropTableWithExistingName`* as `Boolean`) as String

Description
`CreateDBStructureFromXMLSchema` exports the given tables to the specified database. The function returns the SQL statements that were necessary to perform the changes.

See also GetDBStructureList.

Errors
1429 Database selection missing.
1430 Document export failed.

© 2015 Altova GmbH

CreateSchemaDiagram

See also

Method: `CreateSchemaDiagram` (*`nKind`* as `SPYSchemaDefKind`, *`strName`* as String, *`strFile`* as String)

Return Value
None.

Description
The method creates a diagram of the schema type `strName` of kind `nKind` and saves the output file into `strFile`. Note that this functionality is limited to Schema View only. See `Document.CurrentViewMode`. and `SPYViewModes`.

Errors
1400 The object is no longer valid.
1414 Failed to save diagram.
1415 Invalid schema definition type specified.

CurrentViewMode

See also

Method: `CurrentViewMode` as `SPYViewModes`

Description
The property holds the current view mode of the document. See also `Document.SwitchViewMode`.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

DataRoot

See also

Property: `DataRoot` as `XMLData` (read-only)

Description
This property provides access to the document's first XMLData object of type *spyXMLDataElement*. This is typically the root element for all document content data. See XMLSpyDocument.RootElement to get the root element of the whole document including XML prolog data. If the CurrentViewMode is not *spyViewGrid* or *spyViewAuthentic* an UpdateXMLData may be necessary to get access to the latest XMLData.

Errors
1400 The document object is no longer valid.

1407 Invalid address for the return parameter was specified.

DocEditView

See also

Method: `DocEditView` as `DocEditView`

Description
Holds a reference to the current vista Authentic object.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Document needs to be open in authentic view mode.

Encoding

See also

Property: `Encoding` as String

Description
This property provides access to the document's encoding value. However, this property can only be accessed when the document is opened in *spyViewGrid*, *spyViewText* or *spyViewAuthentic*. See CurrentViewMode on how to detect that a document's actual view mode.

This property makes the method SetEncoding obsolete.

Possible values are, for example:

8859-1,
8859-2,
ASCII, ISO-646,
850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.
1416 Operation not supported in current view mode.

EndChanges

See also

© 2015 Altova GmbH

Method: `EndChanges()`

Description
Use the method `EndChanges` to display all changes since the call to `Document.StartChanges`.

Errors
1400 The object is no longer valid.

ExecuteXQuery

See also

Method: `ExecuteXQuery` (*`strXMLFileName`* as String)

Description
Execute the XQuery statements contained in the document of the document object. Either an XQuery execution or an XQuery Update is performed depending on the file extension of the document. Use the XML file specified in the argument as the XML target document that the XQuery document processes.

- If the document has an XQuery file extension as defined in the Options dialog of Authentic Desktop, then an XQuery execution is performed. By default: `.xq`, `.xql`, and `.xquery` are set as XQuery file extensions in Authentic Desktop.
- If the document has an XQuery Update file extension as defined in the Options dialog of Authentic Desktop, then an XQuery Update action is performed. By default: `.xqu` is set as an XQuery Update file extension in Authentic Desktop.

If your XQuery script does not use an XML source, set the parameter `strXMLFileName` to an empty string.

Errors
1400 The document object is no longer valid.
1423 XQuery transformation error.
1424 Not all files required for operation could be loaded. Most likely, the file specified in `strXMLFileName` does not exist or is not valid.

ExportToDatabase

See also

Method: `ExportToDatabase` (*`pFromChild`* as `XMLData`, *`pExportSettings`* as `ExportSettings`, *`pDatabase`* as `DatabaseConnection`)

Description
`ExportToDatabase` exports the XML document starting with the element `pFromChild`. The parameter `pExportSettings` defines the behaviour of the export (see `Application.GetExportSettings`). The parameter `pDatabase` specifies the destination of the export (see `Application.GetDatabaseSettings`). `UpdateXMLData()` might be indirectly needed as you have to pass the `XMLData` as parameter to this function.

Errors

1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.
1416 Error during export.
1429 Database selection missing.
1430 Document export failed.

Example

```
Dim objDoc As Document
Set objDoc = objSpy.ActiveDocument

'set the behaviour of the export with ExportSettings
Dim objExpSettings As ExportSettings
Set objExpSettings = objSpy.GetExportSettings

'set the destination with DatabaseConnection
Dim objDB As DatabaseConnection
Set objDB = objSpy.GetDatabaseSettings

objDB.CreateMissingTables = True
objDB.CreateNew = True
objDB.File = "C:\Export.mdb"

objDoc.ExportToDatabase objDoc.RootElement, objExpSettings, objDB
If Err.Number <> 0 Then
 a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
   "Description: " & Err.Description)
End If
```

ExportToText

See also

Method: ExportToText (*pFromChild* as XMLData, *pExportSettings* as ExportSettings, *pTextSettings* as TextImportExportSettings)

Description

ExportToText exports tabular information from the document starting at pFromChild into one or many text files. Columns of the resulting tables are generated in alphabetical order of the column header names. Use GetExportElementList to learn about the data that will be exported. The parameter pExport Settings defines the specifics for the export. Set the property ExportSettings.ElementList to the - possibly modified - list returned by GetExportElementList to avoid exporting all contained tables. The parameter pText Settings defines the options specific to text export and import. You need to set the property TextImportExportSettings.DestinationFolder before you call ExportToText. UpdateXMLData() might be indirectly needed as you have to pass the XMLData as parameter to this function.

Errors

1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

© 2015 Altova GmbH

1416 Error during export.
1430 Document export failed.

Example

```
' ----------------------------------------------------------
' VBA client code fragment - export document to text files
' ----------------------------------------------------------
Dim objDoc As Document
Set objDoc = objSpy.ActiveDocument

Dim objExpSettings As ExportSettings
Set objExpSettings = objSpy.GetExportSettings
objExpSettings.ElementList = objDoc.GetExportElementList(
        objDoc.RootElement,
        objExpSettings)

Dim objTextExp As TextImportExportSettings
Set objTextExp = objSpy.GetTextExportSettings
objTextExp.HeaderRow = True
objTextExp.DestinationFolder = "C:\Exports"

On Error Resume Next
objDoc.ExportToText objDoc.RootElement, objExpSettings, objTextExp

If Err.Number <> 0 Then
  a = MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) & "Description:
"    & Err.Description)
End If
```

FlattenDTDOrSchema

Method: `FlattenDTDOrSchema` (`sOutputPath` as `String`, `nOutputPathDialogAction` as `SPYDialogAction`)

Parameters

`sOutputPath`
The file path for the newly generated file.

`nOutputPathDialogAction`
Defines the dialog interaction for this call.

Description

`FlattenDTDOrSchema` takes an existing DTD or schema, generates a flattened file, and saves the generated file at the specified location. In the case of DTDs, flattening removes parameter entities and produces a single DTD from a collection of modules; sections marked `IGNORE` are suppressed and unused parameter entities are deleted. When an XML Schema is flattened, (i) the components of all included schemas are added as global components of the active schema, and (ii) included schemas are deleted.

Errors

1400 The object is no longer valid.

1412 Error during conversion.

FullName

See also

Property: `FullName` as String

Description
This property can be used to get or set the full file name - including the path - to where the document gets saved. The validity of the name is not verified before the next save operation.

This property makes the methods GetPathName and SetPathName obsolete.

Errors
1400 The document object is no longer valid.
1402 Empty string has been specified as full file name.

GenerateDTDOrSchema

See also

Method: `GenerateDTDOrSchema` (*`nFormat`* as `SPYDTDSchemaFormat`, *`nValuesList`* as integer, *`nDetection`* as `SPYTypeDetection`, *`nFrequentElements`* as `SPYFrequentElements`)

Parameters
`nFormat`
Sets the schema output format to DTD, or W3C.

`nValuesList`
Generate not more than this amount of enumeration-facets per type. Set to -1 for unlimited.

`nDetection`
Specifies granularity of simple type detection.

`nFrequentElements`
Shall the types for all elements be defined as global? Use that value *spyGlobalComplexType* to define them on global scope. Otherwise, use the value *spyGlobalElements*.

Description
Use this method to automatically generate a DTD or schema for the current XML document.
For a finer tuning of DTD / schema generation, use GenerateDTDOrSchemaEx.
Note that this functionality is not available in ZIP View only. See `Document.CurrentViewMode`. and `SPYViewModes`.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

© 2015 Altova GmbH

GenerateDTDOrSchemaEx

See also

Method: GenerateDTDOrSchemaEx (*objDlg* as DTDSchemaGeneratorDlg) as Document

Description
Use this method to automatically generate a DTD or schema for the current XML document. A DTDSchemaGeneratorDlg object is used to pass information to the schema/DTD generator. The generation process can be configured to allow user interaction or run without further user input. Note that this functionality is not available in ZIP View only. See Document.CurrentViewMode. and SPYViewModes.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

GenerateProgramCode

Method: GenerateProgramCode (*objDlg* as CodeGeneratorDlg)
Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Description
Generate Java, C++ or C# class files from the XML Schema definitions in your document. A CodeGeneratorDlg object is used to pass information to the code generator. The generation process can be configured to allow user interaction or run without further user input.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1421 Feature not available in this edition

GenerateSampleXML

Method: GenerateSampleXML (*objDlg* as GenerateSampleXMLDlg) as Document

Description
Generates a sample XML if the document is a schema or DTD. Use Dialogs.GenerateSampleXMLDlg to get an initialized set of options.

Available with TypeLibrary version 1.5

Errors
1400 The document object is no longer valid.

GenerateSchemaDocumentation

Method: GenerateSchemaDocumentation (*objDlg* as SchemaDocumentationDlg)

Description
Generate documentation for a schema definition file in HTML, MS-Word, or RTF format. The parameter objDlg is used to parameterize the generation process. Use `Dialogs.SchemaDocumentationDlg` to get an initialized set of options. As a minimum, you will need to set the property `SchemaDocumentationDlg.OutputFile` before starting the generation process. Note that this functionality is limited to Schema View only. See `Document.CurrentViewMode` and `SPYViewModes`.

Errors

1400	The document object is no longer valid.
1407	Invalid parameters have been passed or an empty file name has been specified as output target.
1417	The document is not opened in schema view, maybe it is not an '.xsd' file.
1421	Feature is not available in this edition.
1422	Error during generation

GetDBStructureList

See also

Method: `GetDBStructureList` (*`pDatabase`* as `DatabaseConnection`) as `ElementList`

Description
`GetDBStructureList` creates a collection of elements from the Schema document for which tables in the specified database are created. The function returns a collection of `ElementListItems` where the properties `ElementListItem.Name` contain the names of the tables.

See also CreateDBStructureFromXMLSchema.

Errors

1400	The object is no longer valid.
1427	Failed creating parser for the specified XML.
1428	Export of element list failed.
1429	Database selection missing.

GetExportElementList

See also

Method: `GetExportElementList` (*`pFromChild`* as `XMLData`, *`pExportSettings`* as `ExportSettings`) as `ElementList`

Description
`GetExportElementList` creates a collection of elements to export from the document, depending on the settings in `pExportSettings` and starting from the element `pFromChild`. The function returns a collection of `ElementListItems` where the properties `ElementListItem.Name` contain the names of the tables that can be exported from the document. The property `ElementListItem.FieldCount` contains the number of columns in

© 2015 Altova GmbH

the table. The property ElementListItem.RecordCount contains the number of records in the table. The property ElementListItem.ElementKind is unused. UpdateXMLData() might be indirectly needed as you have to pass the XMLData as parameter to this function.

Errors

1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.
1427 Failed creating parser for the specified XML.
1428 Export of element list failed.

GetPathName (obsolete)

Superseded by Document.FullName

```
// ----- javascript sample -----
// instead of:
// strPathName = Application.ActiveDocument.GetPathName();
// use now:
strPathName = Application.ActiveDocument.FullName;
```

See also

Method: GetPathName() as String

Description

The method GetPathName gets the path of the active document.

See also Document.SetPathName (obsolete).

GridView

See also

Property: GridView as GridView

Description

This property provides access to the grid view functionality of the document.

Errors

1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Document needs to be open in enhanced grid view mode.

IsModified

See also

Property: IsModified as Boolean

Description

True if the document is modified.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

IsValid

See also

Method: `IsValid` (*`strError`* as Variant) as Boolean

Return Value
True if the document is valid, false if not.

Description
`IsValid` validates the document against its associated schema or DTD. `strError` gives you the same error message as when you validate the file within the GUI.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.
1408 Unable to validate file.

IsValidEx

Method: `IsValidEx` (*`nXSDVersion`* as `SPYValidateXSDVersion`, *`nErrorLimit`* as `int`, *`nErrorFormat`* as `SPYValidateErrorFormat`, out *`strError`* as Variant) as Boolean

Return Value
True if the document is valid, false if not.

Description
`IsValidEx` validates the document against its associated schema or DTD.

In parameters:
`nXSDVersion` which is an enumeration value of `SPYValidateXSDVersion` that selects the XSD version to validate against.
`nErrorLimit` which is an integer. Values must be `1` to `999`.
`nErrorFormat` which is an enumeration value of `SPYValidateErrorFormat` that selects the XSD version to validate against.

Out parameter:
`strError` is the error message, and is the same as that received when validating the file within the GUI.

Errors
1400 The object is no longer valid.

© 2015 Altova GmbH

1407 Invalid parameter or invalid address for the return parameter was specified.
1408 Unable to validate file.

IsWellFormed

See also

Method: `IsWellFormed` (*`pData`* as `XMLData`, *`bWithChildren`* as `Boolean`, *`strError`* as Variant, *`nErrorPos`* as Variant, *`pBadXMLData`* as Variant) as Boolean

Return Value
True if the document is well formed.

Description
`IsWellFormed` checks the document for well-formedness starting at the element `pData`.

If the document is not well formed, `strError` contains an error message, `nErrorPos` the position in the file and `pBadXMLData` holds a reference to the element which breaks the well-formedness. These out-parameters are defined as VARIANTs to support scripting languages like VBScript.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

Example

See IsValid.

Name

See also

Property: `Name` as String (read-only)

Description
Use this property to retrieve the name - not including the path - of the document file. To change the file name for a document use the property FullName.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

Parent

See also

Property: `Parent` as `Documents` (read-only)

Description
Access the parent of the document object.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

Property: `Parent` as `Application` (read-only)

Path

See also

Property: `Path` as String (read-only)

Description
Use this property to retrieve the path - not including the file name - of the document file. To change the file name and path for a document use the property FullName.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

RootElement

See also

Property: `RootElement` as `XMLData` (read-only)

Description
The property `RootElement` provides access to the root element of the XML structure of the document including the XML prolog data. To access the first element of a document's content navigate to the first child of kind *spyXMLDataElement* or use the Document.DataRoot property. If the CurrentViewMode is not *spyViewGrid* or *spyViewAuthentic* an UpdateXMLData may be necessary to get access to the latest XMLData.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

Save

See also

Method: `Save()`

Description
The method writes any modifications of the document to the associated file. See also `Document.FullName`.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1403 Error when saving file, probably the file name is invalid.

© 2015 Altova GmbH

SaveAs

See also

Method: SaveAs (*strFileName* as String)

Description
Save the document to the file specified. If saving was successful, the FullName property gets set to the specified file name.

Errors
1400 The document object is no longer valid.
1407 An empty file name has been specified.
1403 Error when saving file, probably the file name is invalid.

Saved

See also

Property: Saved as Boolean (read-only)

Description
This property can be used to check if the document has been saved after the last modifications. It returns the negation of IsModified.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

SaveInString

See also

Method: SaveInString (*pData* as XMLData, *bMarked* as Boolean) as String

Parameters
pData
XMLData element to start. Set pData to Document.RootElement if you want to copy the complete file.

bMarked
If bMarked is true, only the elements selected in the grid view are copied.

Return Value
Returns a string with the XML data.

Description
SaveInString starts at the element pData and converts the XMLData objects to a string representation. UpdateXMLData() might be indirectly needed as you have to pass the XMLData as parameter to this function.

Errors

1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

SaveToURL

See also

Method: `SaveToURL` (*`strURL`* as String, *`strUser`* as String, *`strPassword`* as String)

Return Value

Description
`SaveToURL()` writes the document to the URL `strURL`. This method does not set the permanent file path of the document.

Errors
1400 The object is no longer valid.
1402 Invalid URL specified.
1403 Error while saving to URL.

SetActiveDocument

See also

Method: `SetActiveDocument()`

Description
The method sets the document as the active and brings it to the front.

Errors
1400 The object is no longer valid.

SetEncoding (obsolete)

Superseded by Document.Encoding

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetEncoding("UTF-16");
// use now:
Application.ActiveDocument.Encoding = "UTF-16";
```

See also

Method: `SetEncoding` (*`strEncoding`* as String)

Description
`SetEncoding` sets the encoding of the document like the menu item "File/Encoding..." in Authentic Desktop. Possible values for `strEncoding` are, for example:

8859-1,
8859-2,

© 2015 Altova GmbH

ASCII, ISO-646,
850,
1252,
1255,
SHIFT-JIS, MS-KANJI,
BIG5, FIVE,
UTF-7,
UTF-8,
UTF-16

SetExternalIsValid

See also

Method: `SetExternalIsValid` (*`bValid`* as Boolean)

Parameters

`bValid`
Sets the result of an external validation process.

Description
The internal information set by this method is only queried on cancelling the default validation in any OnBeforeValidate handler.

Available with TypeLibrary version 1.5

Errors
1400 The object is no longer valid.

SetPathName (obsolete)

Superseded by Document.FullName

```
// ----- javascript sample -----
// instead of:
// Application.ActiveDocument.SetPathName("C:\\myXMLFiles\\test.xml");
// use now:
Application.ActiveDocument.FullName = "C:\\myXMLFiles\\test.xml";
```

See also

Method: `SetPathName` (*`strPath`* as String)

Description
The method `SetPathName` sets the path of the active document. `SetPathName` only copies the string and does not check if the path is valid. All succeeding save operations are done into this file.

StartChanges

See also

Method: `StartChanges()`

Description
`After StartChanges` is executed Authentic Desktop will not update its editor windows until `Document.EndChanges` is called. This increases performance of complex tasks to the XML structure.

© 2015 Altova GmbH

Errors
1400 The object is no longer valid.

Suggestions

Property: Suggestions as Array

Description
This property contains the last valid user suggestions for this document. The XMLSpy generated suggestions can be modified before they are shown to the user in the OnBeforeShowSuggestions event.

Errors
1400 The object is no longer valid.
1407 Invalid parameter or invalid address for the return parameter was specified.

SwitchViewMode

See also

Method: `SwitchViewMode` (*nMode* as `SPYViewModes`) as Boolean

Return value
Returns true if view mode is switched.

Description
The method sets the current view mode of the document in Authentic Desktop. See also `Document.CurrentViewMode`.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.
1417 Invalid view mode specified.

TextView

See also

Property: TextView as TextView

Description
This property provides access to the text view functionality of the document.

Errors
1400 The object is no longer valid.
1407 Invalid address for the return parameter was specified.

Title

See also

Property: `Title` as String (read-only)

Description
`Title` contains the file name of the document. To get the path and filename of the file use `FullName`.

Errors
1400 The document object is no longer valid.
1407 Invalid address for the return parameter was specified.

TransformXSL

See also

Method: `TransformXSL()`

Description
`TransformXSL` processes the XML document via the associated XSL file. See `Document.AssignXSL` on how to place a reference to a XSL file into the document.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

TransformXSLEx

See also

Method: `TransformXSLEx(nAction as SPYDialogAction)`

Description
`TransformXSLEx` processes the XML document via the associated XSL file. The parameter specifies whether a dialog asking for the result document name should pop up or not. See `Document.AssignXSL` on how to place a reference to a XSL file into the document.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

TransformXSLFO

See also

Method: `TransformXSLFO()`

Description
`TransformXSLFO` processes the XML document via the associated XSLFO file. See `AssignXSLFO` on how to place a reference to a XSLFO file into the document. You need to assign a FOP processor to Authentic Desktop before you can use this method.

Errors
1400 The document object is no longer valid.
1411 Error during transformation process.

© 2015 Altova GmbH

UpdateViews

See also

Method: `UpdateViews()`

Description
To redraw the Enhanced Grid View and the Tree View call `UpdateViews`. This can be important after you changed the `XMLData` structure of a document. This method does not redraw the text view of Authentic Desktop.

Errors
1400 The document object is no longer valid.

UpdateXMLData

See also

Method: `UpdateXMLData()` as Boolean

Description
The XMLData tree is updated from the current view. Please note that this can fail in case of the TextView if the current XML text is not well-formed. This is not necessary if CurrentViewMode is *spyViewGrid* or *spyViewAuthentic* because these views keep the XMLData updated.

Available with TypeLibrary version 1.5

Errors
1400 The document object is no longer valid.

3.2.11 Documents

See also

Properties
Count
Item

Methods
NewAuthenticFile
NewFile
NewFileFromText
OpenAuthenticFile
OpenFile
OpenURL
OpenURLDialog

Description
This object represents the set of documents currently open in Authentic Desktop. Use this object to open further documents or iterate through already opened documents.

Examples

```
' ---------------------------------------
' XMLSpy scripting environment - VBScript
' iterate through open documents
' ---------------------------------------
Dim objDocuments
Set objDocuments = Application.Documents

For Each objDoc In objDocuments
     'do something useful with your document
     objDoc.SetActiveDocument()
Next
```

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// close all open documents
// ---------------------------------------
for (var iter = new Enumerator (Application.Documents);
     ! iter.atEnd();
     iter.moveNext())
{
     // MsgBox ("Closing file " + iter.item().Name);
     iter.item().Close (true);
}
```

Count

See also

Property: Count as long

Description

Count of open documents.

Errors

1600 Invalid `Documents` object
1601 Invalid input parameter

Item

See also

Method: `Item` (*n* as long) as `Document`

Description
Gets the document with the index n in this collection. Index is 1-based.

Errors

1600 Invalid `Documents` object
1601 Invalid input parameter

NewAuthenticFile

See also

Method: `NewAuthenticFile` (*strSPSPath* as String, *strXMLPath* as String) as `Document`

Parameters
`strSPSPath`
The path to the SPS document.

`strXMLPath`
The new XML document name.

Return Value
The method returns the new document.

Description
NewAuthenticFile creates a new XML file and opens it in Authentic View using SPS design strSPSPath.

NewFile

See also

Method: `NewFile` (*strFile* as String, *strType* as String) as `Document`

Parameters
`strFile`
Full path of new file.

`strType`
Type of new file as string (i.e. "xml", "xsd", ...)

Return Value
Returns the new file.

Description
`NewFile` creates a new file of type `strType` (i.e. "xml"). The newly created file is also the ActiveDocument.

NewFileFromText

See also

Method: `NewFileFromText` (*`strText`* as String, *`strType`* as String) as `Document`

Parameters
`strText`
The content of the new document in plain text.

`strType`
Type of the document to create (i.e. "xml").

Return Value
The method returns the new document.

Description
NewFileFromText creates a new document with strText as its content.

OpenAuthenticFile

See also

Method: `OpenAuthenticFile` (*`strSPSPath`* as String, *`strXMLPath`* as String) as `Document`

Parameters
`strSPSPath`
The path to the SPS document.

`strXMLPath`
The path to the XML document (can be empty).

Return Value
The method returns the new document.

Description
OpenAuthenticFile opens an XML file or database in Authentic View using SPS design strSPSPath.

OpenFile

See also

Method: `OpenFile` (*`strPath`* as String, *`bDialog`* as Boolean) as `Document`

© 2015 Altova GmbH

Parameters
`strPath`
Path and file name of file to open.

`bDialog`
Show dialogs for user input.

Return Value
Returns the opened file on success.

Description
`OpenFile` opens the file `strPath`. If `bDialog` is TRUE, a file-dialog will be displayed.

Example

```
Dim objDoc As Document
Set objDoc = objSpy.Documents.OpenFile(strFile, False)
```

OpenURL

See also

Method: `OpenURL` (*`strURL`* as String, *`nURLType`* as `SPYURLTypes`, *`nLoading`* as `SPYLoading`, *`strUser`* as String, *`strPassword`* as String) as `Document`

Parameters
`strURL`
URL to open as document.

`nURLType`
Type of document to open. Set to -1 for auto detection.

`nLoading`
Set `nLoading` to 0 (zero) if you want to load it from cache or proxy. Otherwise set `nLoading` to 1.

`strUser`
Name of the user if required. Can be empty.

`strPassword`
Password for authentification. Can be empty.

Return Value
The method returns the opened document.

Description
`OpenURL` opens the URL strURL.

OpenURLDialog

See also

Method: `OpenURLDialog` (*`strURL`* as String, *`nURLType`* as `SPYURLTypes`, *`nLoading`* as `SPYLoading`, *`strUser`* as String, *`strPassword`* as String) as `Document`

Parameters

`strURL`
URL to open as document.

`nURLType`
Type of document to open. Set to -1 for auto detection.

`nLoading`
Set `nLoading` to 0 (zero) if you want to load it from cache or proxy. Otherwise set `nLoading` to 1.

`strUser`
Name of the user if required. Can be empty.

`strPassword`
Password for authentification. Can be empty.

Return Value
The method returns the opened document.

Description
`OpenURLDialog` displays the "open URL" dialog to the user and presets the input fields with the given parameters.

© 2015 Altova GmbH

3.2.12 DTDSchemaGeneratorDlg

See also

Properties and Methods

Standard automation properties
Application
Parent

DTDSchemaFormat
ValueList
TypeDetection
FrequentElements
MergeAllEqualNamed
ResolveEntities
AttributeTypeDefinition
GlobalAttributes
OnlyStringEnums
MaxEnumLength
OutputPath
OutputPathDialogAction

Description
Use this object to configure the generation of a schema or DTD. The method GenerateDTDOrSchemaEx expects a DTDSchemaGeneratorDlg as parameter to configure the generation as well as the associated user interactions.

Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

AttributeTypeDefinition

Property: AttributeTypeDefinition as SPYAttributeTypeDefinition

Description
Specifies how attribute definitions get merged.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

DTDSchemaFormat

Property: DTDSchemaFormat as `SPYDTDSchemaFormat`

Description
Sets the schema output format to DTD, or W3C.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

FrequentElements

Property: FrequentElements as `SPYFrequentElements`

Description
Shall the types for all elements be defined as global? Use that value *spyGlobalComplexType* to define them on global scope. Otherwise, use the value *spyGlobalElements*.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

GlobalAttributes

Property: GlobalAttributes as Boolean

Description
Shall attributes with same name and type be resolved globally?

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

MaxEnumLength

Property: MaxEnumLength as Integer

Description
Specifies the maximum number of characters allowed for enumeration names. If one value is longer than this, no enumeration will be generated.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

MergeAllEqualNamed

Property: MergeAllEqualNamed as Boolean

Description
Shall types of all elements with the same name be merged into one type?

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

OnlyStringEnums

Property: OnlyStringEnums as Boolean

Description
Specifies if enumerations will be created only for plain strings or all types of values.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

OutputPath

Property: OutputPath as String

Description
Selects the file name for the generated schema/DTD.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

OutputPathDialogAction

Property: OutputPathDialogAction as `SPYDialogAction`

Description
Defines how the sub-dialog for selecting the schema/DTD output path gets handled. Set this value to *spyDialogUserInput(2)* to show the dialog with the current value of the OutputPath property as default. Use *spyDialogOK(0)* to hide the dialog from the user.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

Parent

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

ResolveEntities

Property: ResolveEntities as Boolean

Description
Shall all entities be resolved before generation starts? If yes, an info-set will be built.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

TypeDetection

Property: TypeDetection as `SPYTypeDetection`

Description
Specifies granularity of simple type detection.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

ValueList

Property: ValueList as Integer

Description
Generate not more than this amount of enumeration-facets per type. Set to -1 for unlimited.

Errors
3000 The object is no longer valid.
3001 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

3.2.13 ElementList

See also

Properties
Count
Item

Methods
RemoveElement

Description
Element lists are used for different purposes during export and import of data. Depending on this purpose, different properties of ElementListItem are used.

It can hold
- a list of table names returned by a call to Application.GetDatabaseTables,
- a list of field names retuned by a call to Application.GetDatabaseImportElementList or Application.GetTextImportElementList,
- a field name filter list used in Application.ImportFromDatabase and Application.ImportFromText,
- a list of table names and counts for their rows and columns as returned by calls to GetExportElementList or
- a field name filter list used in Document.ExportToDatabase and Document.ExportToText.

Count

See also

Property: Count as long (read-only)

Description
Count of elements in this collection.

Item

See also

Method: Item(n as long) as ElementListItem

Description
Gets the element with the index n from this collection. The first item has index 1.

RemoveElement

See also

Method: RemoveElement(Index as long)

Description

`RemoveElement` removes the element Index from the collection. The first Item has index 1.

© 2015 Altova GmbH

3.2.14 ElementListItem

See also

Properties
Name

ElementKind

FieldCount
RecordCount

Description
An element in an ElementList. Usage of its properties depends on the purpose of the element list. For details see ElementList.

ElementKind

See also

Property: ElementKind as SPYXMLDataKind

Description
Specifies if a field should be imported as XML element (data value of spyXMLDataElement) or attribute (data value of spyXMLDataAttr).

FieldCount

See also

Property: FieldCount as long (read-only)

Description
Count of fields (i.e. columns) in the table described by this element. This property is only valid after a call to Document.GetExportElementList.

Name

See also

Property: Name as String (read-only)

Description
Name of the element. This is either the name of a table or a field, depending on the purpose of th element list.

RecordCount

See also

Property: RecordCount as long (read-only)

Description
Count of records (i.e. rows) in the table described by this element. This property is only valid after a call to `Document.GetExportElementList`.

© 2015 Altova GmbH

3.2.15 ExportSettings

See also

Properties

ElementList

EntitiesToText

ExportAllElements
SubLevelLimit

FromAttributes
FromSingleSubElements
FromTextValues

CreateKeys
IndependentPrimaryKey

Namespace

ExportCompleteXML
StartFromElement

Description
ExportSettings contains options used during export of XML data to a database or text file.

CreateKeys

See also

Property: CreateKeys as Boolean

Description
This property turns creation of keys (i.e. primary key and foreign key) on or off. Default is True.

ElementList

See also

Property: ElementList as ElementList

Description
Default is empty list. This list of elements defines which fields will be exported. To get the list of available fields use Document.GetExportElementList. It is possible to prevent exporting columns by removing elements from this list with ElementList.RemoveElement before passing it to Document.ExportToDatabase or Document.ExportToText.

EntitiesToText

See also

Property: `EntitiesToText` as Boolean

Description
Defines if XML entities should be converted to text or left as they are during export. Default is True.

ExportAllElements

See also

Property: `ExportAllElements` as Boolean

Description
If set to `true`, all elements in the document will be exported. If set to `false`, then `ExportSettings.SubLevelLimit` is used to restrict the number of sub levels to export. Default is `true`.

ExportCompleteXML

See also

Property: ExportCompleteXML as Boolean

Description
Defines whether the complete XML is exported or only the element specified by StartFromElement and its children. Default is True.

FromAttributes

See also

Property: `FromAttributes` as Boolean

Description
`Set FromAttributes` to false if no export data should be created from attributes. Default is True.

FromSingleSubElements

See also

Property: `FromSingleSubElements` as Boolean

Description
`Set FromSingleSubElements` to false if no export data should be created from elements. Default is True.

FromTextValues

See also

Property: `FromTextValues` as Boolean

© 2015 Altova GmbH

Description
`Set FromTextValues` to false if no export data should be created from text values. Default is True.

IndependentPrimaryKey

See also

Property: `IndependentPrimaryKey` as Boolean

Description
Turns creation of independent primary key counter for every element on or off. If `ExportSettings.CreateKeys` is False, this property will be ignored. Default is True.

Namespace

See also

Property: `Namespace` as `SPYExportNamespace`

Description
The default setting removes all namespace prefixes from the element names. In some database formats the colon is not a legal character. Default is `spyNoNamespace`.

StartFromElement

See also

Property: StartFromElement as String

Description
Specifies the start element for the export. This property is only considered when ExportCompleteXML is false.

SubLevelLimit

See also

Property: `SubLevelLimit` as Integer

Description
Defines the number of sub levels to include for the export. Default is 0. This property is ignored if `ExportSettings.ExportAllElements` is `true`.

3.2.16 FileSelectionDlg

See also

Properties and Methods

Standard automation properties
Application
Parent

Dialog properties
FullName

Acceptance or cancellation of action that caused event
DialogAction

Description
The dialog object allows you to receive information about an event and pass back information to the event handler in the same way as with a user dialog. Use the FileSelectionDlg.FullName to select or modify the file path and set the FileSelectionDlg.DialogAction property to cancel or agree with the action that caused the event.

Application

See also

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

DialogAction

Property: DialogAction as SPYDialogAction

Description
If you want your script to perform the file selection operation without any user interaction necessary, simulate user interaction by either setting the property to *spyDialogOK(0)* or *spyDialogCancel(1)*.
To allow your script to fill in the default values but let the user see and react on the dialog, use the value *spyDialogUserInput(2)*. If you receive a FileSelectionDlg object in an event handler, *spyDialogUserInput(2)* is not supported and will be interpreted as *spyDialogOK(0)*.

Errors
2400 The object is no longer valid.
2401 Invalid value for dialog action or invalid address for the return parameter was specified.

© 2015 Altova GmbH

FullName

Property: `FullName` as String

Description
Access the full path of the file the gets selected by the dialog. Most events that pass a FileSelectionDlg object to you allow you modify this value and thus influence the action that caused the event (e.g. load or save to a different location).

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

Parent

See also

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
2400 The object is no longer valid.
2401 Invalid address for the return parameter was specified.

3.2.17 FindInFilesDlg

See also

Properties and Methods

Standard automation properties
Application
Parent

Find
RegularExpression
Replace
DoReplace
ReplaceOnDisk
MatchWholeWord
MatchCase
SearchLocation
StartFolder
IncludeSubfolders
SearchInProjectFilesDoExternal
FileExtension
AdvancedXMLSearch
XMLElementNames
XMLElementContents
XMLAttributeNames
XMLAttributeContents
XMLComments
XMLCData
XMLPI
XMLRest
ShowResult

Description
Use this object to configure the search (or replacement) for strings in files. The method FindInFiles expects a FindInFilesDlg as parameter.

AdvancedXMLSearch

Property: AdvancedXMLSearch as Boolean

Description
Specifies if the XML search properties (XMLElementNames, XMLElementContents, XMLAttributeNames, XMLAttributeContents, XMLComments, XMLCData, XMLPI and XMLRest) are considered. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

Application

Property: `Application` as `Application` (read-only)

Description
Access the Authentic Desktop application object.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

DoReplace

Property: DoReplace as Boolean

Description
Specifies if the matched string is replaced by the string defined in Replace. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

FileExtension

Property: FileExtension as String

Description
Specifies the file filter of the files that should be considered during the search. Multiple file filters must be delimited with a semicolon (eg: *.xml;*.dtd;a*.xsd). Use the wildcards * and ? to define the file filter.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

Find

Property: Find as String

Description
Specifies the string to search for.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

IncludeSubfolders

Property: IncludeSubfolders as Boolean

Description

Specifies if subfolders are searched too. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

MatchCase

Property: MatchCase as Boolean

Description
Specifies if the search is case sensitive. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

MatchWholeWord

Property: MatchWholeWord as Boolean

Description
Specifies whether the whole word or just a part of it must match. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

Parent

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

RegularExpression

Property: RegularExpression as Boolean

Description
Specifies if Find contains a regular expression. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

Replace

Property: Replace as String

Description
Specifies the replacement string. The matched string is only replaced if DoReplace is set true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

ReplaceOnDisk

Property: ReplaceOnDisk as Boolean

Description
Specifies if the replacement is done directly on disk. The modified file is not opened. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

SearchInProjectFilesDoExternal

Property: SearchInProjectFilesDoExternal as Boolean

Description
Specifies if the external folders in the open project are searched, when a project search is performed. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

SearchLocation

Property: SearchLocation as SPYFindInFilesSearchLocation

Description
Specifies the location of the search. The default is spyFindInFiles_Documents.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

ShowResult

Property: ShowResult as Boolean

Description
Specifies if the result is displayed in the Find in Files output window. The default is false.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

StartFolder

Property: StartFolder as String

Description
Specifies the folder where the disk search starts.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLAttributeContents

Property: XMLAttributeContents as Boolean

Description
Specifies if attribute contents are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLAttributeNames

Property: XMLAttributeNames as Boolean

Description
Specifies if attribute names are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLCData

Property: XMLCData as Boolean

Description
Specifies if CData tags are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLComments

Property: XMLComments as Boolean

© 2015 Altova GmbH

Description
Specifies if comments are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLElementContents

Property: XMLElementContents as Boolean

Description
Specifies if element contents are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLElementNames

Property: XMLElementNames as Boolean

Description
Specifies if element names are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLPI

Property: XMLPI as Boolean

Description
Specifies if XML processing instructions are searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

XMLRest

Property: XMLRest as Boolean

Description
Specifies if the rest of the XML (which is not covered by the other XML search properties) is searched when AdvancedXMLSearch is true. The default is true.

Errors
3500 The object is no longer valid.
3501 Invalid address for the return parameter was specified.

3.2.18 FindInFilesResult

See also

Properties and Methods

Standard automation properties
Application
Parent

Count
Item

Path
Document

Description
This object represents a file that matched the search criteria. It contains a list of FindInFilesResultMatch objects that describe the matching position.

Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

Count

Property: Count as long (read-only)

Description
Count of elements in this collection.

Document

Property: Path as Document (read-only)

Description
This property returns the Document object if the matched file is already open in XMLSpy.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

Item

Method: `Item(n` as long) as FindInFilesResultMatch

Description
Gets the element with the index `n` from this collection. The first item has index 1.

Parent

Property: `Parent` as FindInFilesResults (read-only)

Description
Access the parent of the object.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

Path

Property: Path as String (read-only)

Description
Returns the path of the file that matched the search criteria.

Errors
3700 The object is no longer valid.
3701 Invalid address for the return parameter was specified.

3.2.19 FindInFilesResultMatch

See also

Properties and Methods

Standard automation properties
Application
Parent

Line
Position
Length
LineText
Replaced

Description
Contains the exact position in the file of the matched string.

Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

Length

Property: Length as Long (read-only)

Description
Returns the length of the matched string.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

Line

Property: Line as Long (read-only)

Description
Returns the line number of the match. The line numbering starts with 0.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

LineText

Property: LineText as String (read-only)

Description
Returns the text of the line.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

Parent

Property: `Parent` as FindInFilesResult (read-only)

Description
Access the parent of the object.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

Position

Property: Position as Long (read-only)

Description
Returns the start position of the match in the line. The position numbering starts with 0.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

Replaced

Property: Replaced as Boolean (read-only)

Description
True if the matched string was replaced.

Errors
3800 The object is no longer valid.
3801 Invalid address for the return parameter was specified.

3.2.20 FindInFilesResults

See also

Properties and Methods

Standard automation properties
Application
Parent

Count
Item

Description
This is the result of the FindInFiles method. It is a list of FindInFilesResult objects.

Application

Property: `Application` as `Application` (read-only)

Description
Access the Authentic Desktop application object.

Errors
3600 The object is no longer valid.
3601 Invalid address for the return parameter was specified.

Count

Property: `Count` as long (read-only)

Description
Count of elements in this collection.

Item

Method: `Item(n` as long) as FindInFilesResult

Description
Gets the element with the index `n` from this collection. The first item has index 1.

Parent

Property: `Parent` as Application (read-only)

Description
Access the parent of the object.

Errors
3600 The object is no longer valid.
3601 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

3.2.21 GenerateSampleXMLDlg

See also

Properties and Methods

Standard automation properties
Application
Parent

NonMandatoryAttributes
NonMandatoryElements
RepeatCount
FillAttributesWithSampleData
FillElementsWithSampleData
ContentOfNillableElementsIsNonMandatory
TryToUseNonAbstractTypes
SchemaOrDTDAssignment
LocalNameOfRootElement
NamespaceURIOfRootElement
OptionsDialogAction

Properties that are no longer supported
TakeFirstChoice - obsolete
FillWithSampleData - obsolete
Optimization - obsolete

Description
Used to set the parameters for the generation of sample XML instances based on a W3C schema or DTD.

Application

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

ChoiceMode

Property: ChoiceMode as SPYSampleXMLGenerationChoiceMode

Description
Specifies which elements will be generated.

Errors

2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

ConsiderSampleValueHints

Property: `ConsiderSampleValueHints` as Boolean

Description
Selects whether to use `SampleValueHints` or not.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

ContentOfNillableElementsIsNonMandatory

Property: `ContentOfNillableElementsIsNonMandatory` as Boolean

Description
If true, the contents of elements that are nillable will not be treated as mandatory.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

FillAttributesWithSampleData

Property: `FillAttributesWithSampleData` as Boolean

Description
If true, attributes will have sample content.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

FillElementsWithSampleData

Property: `FillElementsWithSampleData` as Boolean

Description
If true, elements will have sample content.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

FillWithSampleData - obsolete

Property: `FillWithSampleData` as Boolean

Description
Do no longer access this property. Use `FillAttributesWithSampleData` and `FillElementsWithSampleData`, instead.

Errors
0001 The property is no longer accessible.

LocalNameOfRootElement

Property: `LocalNameOfRootElement` as String

Description
Specifies the local name of the root element for the generated sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

NamespaceURIOfRootElement

Property: `NamespaceURIOfRootElement` as String

Description
Specifies the namespace URI of the root element for the generated sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

NonMandatoryAttributes

Property: `NonMandatoryAttributes` as Boolean

Description
If true attributes which are not mandatory are created in the sample XML instance file.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

NonMandatoryElements

Property: `NonMandatoryElements` as Boolean

Description
If `true`, elements which are not mandatory are created in the sample XML instance file.

Errors
2200 The object is no longer valid.
2201 Invalid address was specified for the return parameter.

Optimization - obsolete

Property: `Optimization` as `SPYSampleXMLGenerationOptimization`

Description
Do not use this property any longer. Use ChoiceMode and NonMandatoryElements.

Errors
0001 The property is no longer accessible.

OptionsDialogAction

Property: `OptionsDialogAction` as `SPYDialogAction`

Description
To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors
2200 The object is no longer valid.
2201 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

Parent

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

RepeatCount

Property: `RepeatCount` as long

Description
Number of elements to create for repeated types.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

SampleValueHints

Property: `SampleValueHints` as `SPYSampleXMLGenerationSampleValueHints`

Description
Specifies how to select data for the generated sample file.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

SchemaOrDTDAssignment

Property: `SchemaOrDTDAssignment` as `SPYSampleXMLGenerationSchemaOrDTDAssignment`

Description
Specifies in which way a reference to the related schema or DTD - which is this document - will be generated into the sample XML.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

TakeFirstChoice - obsolete

Property: `TakeFirstChoice` as Boolean

Description
Do no longer use this property.

Errors
0001 The property is no longer accessible.

TryToUseNonAbstractTypes

Property: `TryToUseNonAbstractTypes` as Boolean

Description
If true, tries to use a non-abstract type for xsi:type, if element has an abstract type.

Errors
2200 The object is no longer valid.
2201 Invalid address for the return parameter was specified.

3.2.22 GridView

See also

Methods
Deselect
Select

SetFocus

Properties
CurrentFocus

IsVisible

Description
GridView Class

Events

OnBeforeDrag

See also

Event: `OnBeforeDrag() as Boolean`

XMLSpy scripting environment - VBScript:
```
Function On_BeforeDrag()
     ' On_BeforeStartEditing = False  ' to prohibit dragging
End Function
```

XMLSpy scripting environment - JScript:
```
function On_BeforeDrag()
{
     // return false;  /* to prohibit dragging */
}
```

XMLSpy IDE Plugin:
```
IXMLSpyPlugIn.OnEvent (4, ...)   // nEventId = 4
```

Description
This event gets fired on an attempt to drag an XMLData element on the grid view. Return *false* to prevent dragging the data element to a different position.

OnBeforeDrop

See also

Event: `OnBeforeDrop(objXMLData as XMLData) as Boolean`

XMLSpy scripting environment - VBScript:

```
Function On_BeforeDrop(objXMLData)
     ' On_BeforeStartEditing = False  ' to prohibit dropping
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeDrop(objXMLData)
{
     // return false;  /* to prohibit dropping */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (5, ...)   // nEventId = 5
```

Description

This event gets fired on an attempt to drop a previously dragged XMLData element on the grid view. Return *false* to prevent the data element to be moved from its original position to the drop destination position.

OnBeforeStartEditing

See also

Event: `OnBeforeStartEditing(objXMLData as XMLData, bEditingName as Boolean)as Boolean`

XMLSpy scripting environment - VBScript:

```
Function On_BeforeStartEditing(objXMLData, bEditingName)
     ' On_BeforeStartEditing = False  ' to prohibit editing the field
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeStartEditing(objXMLData, bEditingName)
{
     // return false;  /* to prohibit editing the field */
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (1, ...)   // nEventId = 1
```

Description

This event gets fired before the editing mode for a grid cell gets entered. If the parameter *bEditingName* is true, the name part of the element will be edited, it its value is false, the value part will be edited.

OnEditingFinished

See also

Event: `OnEditingFinished(objXMLData as XMLData, bEditingName as Boolean)`

XMLSpy scripting environment - VBScript:

```
Function On_EditingFinished(objXMLData, bEditingName)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_EditingFinished(objXMLData, bEditingName)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (2, ...)   // nEventId = 2
```

Description

This event gets fired when the editing mode of a grid cell gets left. The parameter *bEditingName* specifies if the name part of the element has been edited.

OnFocusChanged

See also

Event: `OnFocusChanged(objXMLData as XMLData, bSetFocus as Boolean, bEditingName as Boolean)`

XMLSpy scripting environment - VBScript:

```
Function On_FocusChanged(objXMLData, bSetFocus, bEditingName)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_FocusChanged(objXMLData, bSetFocus, bEditingName)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (3, ...)   // nEventId = 3
```

Description

This event gets fired whenever a grid cell receives or looses the cursor focus. If the parameter *bEditingName* is *true*, focus of the name part of the grid element has changed. Otherwise, focus of the value part has changed.

CurrentFocus

See also

Property: `CurrentFocus as XMLData`

Description

© 2015 Altova GmbH

Holds the XML element with the current focus. This property is read-only.

Deselect

See also

Method: `Deselect(pData as XMLData)`

Description
Deselects the element `pData` in the grid view.

IsVisible

See also

Property: `IsVisible` as Boolean

Description
True if the grid view is the active view of the document. This property is read-only.

Select

See also

Method: `Select (pData as XMLData)`

Description
Selects the XML element `pData` in the grid view.

SetFocus

See also

Method: `SetFocus (pFocusData as XMLData)`

Description
Sets the focus to the element `pFocusData` in the grid view.

3.2.23 SchemaDocumentationDlg

See also

Properties and Methods

Standard automation properties
Application
Parent

Interaction and visibility properties
OutputFile
OutputFileDialogAction
OptionsDialogAction
ShowProgressBar
ShowResult

Document generation options and methods
OutputFormat
UseFixedDesign
SPSFile
EmbedDiagrams
DiagramFormat
MultipleOutputFiles
EmbedCSSInHTML
CreateDiagramsFolder
GenerateRelativeLinks

IncludeAll
IncludeIndex
IncludeGlobalAttributes
IncludeGlobalElements
IncludeLocalAttributes
IncludeLocalElements
IncludeGroups
IncludeComplexTypes
IncludeSimpleTypes
IncludeAttributeGroups
IncludeRedefines
IncludeReferencedSchemas

AllDetails
ShowDiagram
ShowNamespace
ShowType
ShowChildren
ShowUsedBy
ShowProperties
ShowSingleFacets
ShowPatterns
ShowEnumerations
ShowAttributes
ShowIdentityConstraints

© 2015 Altova GmbH

ShowAnnotations
ShowSourceCode

Description
This object combines all options for schema document generation as they are available through user interface dialog boxes in Authentic Desktop. The document generation options are initialized with the values used during the last generation of schema documentation. However, before using the object you have to set the SetOutputFile property to a valid file path. Use OptionsDialogAction, OutputFileDialogAction and ShowProgressBar to specify the level of user interaction desired. You can use IncludeAll and AllDetails to set whole option groups at once or the individual properties to operate on a finer granularity.

AllDetails

See also

Method: AllDetails (i_bDetailsOn as Boolean)

Description
Use this method to turn all details options on or off.

Errors
2900 The object is no longer valid.

Application

See also

Property: Application as Application (read-only)

Description
Access the Authentic Desktop application object.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

CreateDiagramsFolder

See also

Property: CreateDiagramsFolder as Boolean

Description
Set this property to true, to create a directory for the created images. Otherwise the diagrams will be created next to the documentation. This property is only available when the diagrams are not embedded. The default for the first run is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

DiagramFormat

See also

Property: `DiagramFormat` as SPYImageKind

Description
This property specifies the generated diagram image type. This property is not available for HTML documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is PNG.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

EmbedCSSInHTML

See also

Property: `EmbedCSSInHTML` as `Boolean`

Description
Set this property to `true`, to embed the CSS data in the generated HTML document. Otherwise a separate file will be created and linked. This property is only available for HTML documentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

EmbedDiagrams

See also

Property: `EmbedDiagrams` as `Boolean`

Description
Set this property to `true`, to embed the diagrams in the generated document. This property is not available for HTML documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

GenerateRelativeLinks

See also

Property: GenerateRelativeLinks as `Boolean`

Description
Set this property to `true`, to create relative paths to local files. This property is not available for HTML documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeAll

See also

Method: `IncludeAll` (`i_bInclude` as `Boolean`)

Description
Use this method to mark or unmark all include options.

Errors
2900 The object is no longer valid.

IncludeAttributeGroups

See also

Property: `IncludeAttributeGroups` as `Boolean`

Description
Set this property to `true`, to include attribute groups in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeComplexTypes

See also

Property: `IncludeComplexTypes` as `Boolean`

Description

Set this property to `true`, to include complex types in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeGlobalAttributes

See also

Property: `IncludeGlobalAttributes` as `Boolean`

Description

Set this property to `true`, to include global attributes in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeGlobalElements

See also

Property: `IncludeGlobalElements` as `Boolean`

Description

Set this property to `true`, to include global elements in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeGroups

See also

Property: `IncludeGroups` as `Boolean`

Description

Set this property to `true`, to include groups in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors

2900 The object is no longer valid.

© 2015 Altova GmbH

2901 Invalid address for the return parameter was specified.

IncludeIndex

See also

Property: `IncludeIndex` as `Boolean`

Description
Set this property to `true`, to include an index in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeLocalAttributes

See also

Property: `IncludeLocalAttributes` as `Boolean`

Description
Set this property to `true`, to include local attributes in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeLocalElements

See also

Property: `IncludeLocalElements` as `Boolean`

Description
Set this property to `true`, to include local elements in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeRedefines

See also

Property: `IncludeRedefines` as `Boolean`

Description
Set this property to `true`, to include redefines in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeReferencedSchemas

See also

Property: `IncludeReferencedSchemas` as `Boolean`

Description
Set this property to `true`, to include referenced schemas in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

IncludeSimpleTypes

See also

Property: `IncludeSimpleTypes` as `Boolean`

Description
Set this property to `true`, to include simple types in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

MultipleOutputFiles

See also

Property: `MultipleOutputFiles` as Boolean

Description
Set this property to `true`, to split the documentation files. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is false.

© 2015 Altova GmbH

Errors

2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

OptionsDialogAction

See also

Property: `OptionsDialogAction` as `SPYDialogAction`

Description

To allow your script to fill in the default values and let the user see and react on the dialog, set this property to the value *spyDialogUserInput(2)*. If you want your script to define all the options in the schema documentation dialog without any user interaction necessary, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors

2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

OutputFile

See also

Property: `OutputFile` as `String`

Description

Full path and name of the file that will contain the generated documentation. In case of HTML output, additional '.png' files will be generated based on this filename. The default value for this property is an empty string and needs to be replaced before using this object in a call to `Document.GenerateSchemaDocumentation`.

Errors

2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

OutputFileDialogAction

See also

Property: `OutputFileDialogAction` as `SPYDialogAction`

Description

To allow the user to select the output file with a file selection dialog, set this property to *spyDialogUserInput(2)*. If the value stored in OutputFile should be taken and no user interaction should occur, use *spyDialogOK(0)*. Default is *spyDialogOK*.

Errors

2900 The object is no longer valid.

2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

OutputFormat

See also

Property: `OutputFormat` as `SPYSchemaDocumentationFormat`

Description
Defines the kind of documentation that will be generated: HTML (value=`0`), MS-Word (value=`1`), or RTF (value=`2`). The property gets initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is HTML.

Errors
2900 The object is no longer valid.
2901 Invalid value has been used to set the property.
Invalid address for the return parameter was specified.

Parent

See also

Property: `Parent` as `Dialogs` (read-only)

Description
Access the parent of the object.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowAnnotations

See also

Property: `ShowAnnotations` as `Boolean`

Description
Set this property to `true`, to show the annotations to a type definition in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowAttributes

See also

Property: `ShowAttributes` as `Boolean`

© 2015 Altova GmbH

Description
Set this property to `true`, to show the type definitions attributes in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowChildren

See also

Property: `ShowChildren` as `Boolean`

Description
Set this property to `true`, to show the children of a type definition as links in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowDiagram

See also

Property: `ShowDiagram` as `Boolean`

Description
Set this property to `true`, to show type definitions as diagrams in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowEnumerations

See also

Property: `ShowEnumerations` as `Boolean`

Description
Set this property to `true`, to show the enumerations contained in a type definition in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowIdentityConstraints

See also

Property: `ShowIdentityConstraints` as `Boolean`

Description
Set this property to `true`, to show a type definitions identity constraints in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowNamespace

See also

Property: `ShowNamespace` as `Boolean`

Description
Set this property to `true`, to show the namespace of type definitions in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowPatterns

See also

Property: `ShowPatterns` as `Boolean`

Description
Set this property to `true`, to show the patterns of a type definition in the schema documentation. The property is initialized with the value used during the last call to `Document.GenerateSchemaDocumentation`. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

ShowProgressBar

See also

Property: ShowProgressBar as Boolean

Description
Set this property to true, to make the window showing the document generation progress visible. Use false, to hide it. Default is false.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowProperties

See also

Property: ShowProperties as Boolean

Description
Set this property to true, to show the type definition properties in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowResult

See also

Property: ShowResult as Boolean

Description
Set this property to true, to automatically open the resulting document when generation was successful. HTML documentation will be opened in Authentic Desktop. To show Word documentation, MS-Word will be started. The property gets initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowSingleFacets

See also

Property: ShowSingleFacets as Boolean

Description
Set this property to true, to show the facets of a type definition in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowSourceCode

See also

Property: ShowSourceCode as Boolean

Description
Set this property to true, to show the XML source code for type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowType

See also

Property: ShowType as Boolean

Description
Set this property to true, to show the type of type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

ShowUsedBy

See also

© 2015 Altova GmbH

Property: ShowUsedBy as Boolean

Description
Set this property to true, to show the used-by relation for type definitions in the schema documentation. The property is initialized with the value used during the last call to Document.GenerateSchemaDocumentation. The default for the first run is true.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

SPSFile

See also

Property: SPSFile as String

Description
Full path and name of the SPS file that will be used to generate the documentation.

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

UseFixedDesign

See also

Property: UseFixedDesign as Boolean

Description
Specifies whether the documentation should be created with a fixed design or with a design specified by a SPS file (which requires StyleVision).

Errors
2900 The object is no longer valid.
2901 Invalid address for the return parameter was specified.

3.2.24 SpyProject

See also

Methods
CloseProject
SaveProject
SaveProjectAs

Properties
RootItems
ProjectFile

Description
SpyProject Class

CloseProject

See also

Declaration: CloseProject(*bDiscardChanges* as Boolean, *bCloseFiles* as Boolean, *bDialog* as Boolean)

Parameters
bDiscardChanges
Set bDiscardChanges to FALSE if you want to save the changes of the open project files and the project.

bCloseFiles
Set bCloseFiles to TRUE to close all open project files.

bDialog
Show dialogs for user input.

Description
CloseProject closes the current project.

ProjectFile

See also

Declaration: ProjectFile as String

Description
Path and filename of the project.

RootItems

See also

Declaration: RootItems as SpyProjectItems

© 2015 Altova GmbH

Description
Root level of collection of project items.

SaveProject

See also

Declaration: `SaveProject`

Description
`SaveProject` saves the current project.

SaveProjectAs

See also

Declaration: `SaveProjectAs` (`strPath` as String, `bDialog` as Boolean)

Parameters
`strPath`
Full path with file name of new project file.

`bDialog`
If `bDialog` is TRUE, a file-dialog will be displayed.

Description
`SaveProjectAs` stores the project data into a new location.

3.2.25 SpyProjectItem

See also

Methods
Open

Properties
ChildItems
ParentItem
FileExtensions
ItemType
Name
Path
ValidateWith
XMLForXSLTransformation
XSLForXMLTransformation
XSLTransformationFileExtension
XSLTransformationFolder

Description
SpyProjectItem Class

ChildItems

See also

Declaration: ChildItems as SpyProjectItems

Description
If the item is a folder, ChildItems is the collection of the folder content.

FileExtensions

See also

Declaration: FileExtensions as String

Description
Used to set the file extensions if the project item is a folder.

ItemType

See also

Declaration: ItemType as SPYProjectItemTypes

Description
This property is read-only.

© 2015 Altova GmbH

Name

See also

Declaration: `Name` as String

Description
Name of the project item. This property is read-only.

Open

See also

Declaration: `Open` as `Document`

Return Value
The project item opened as document.

Description
Opens the project item.

ParentItem

See also

Declaration: `ParentItem` as `SpyProjectItem`

Description
Parent item of the current project item. Can be NULL (Nothing) if the project item is a top-level item.

Path

See also

Declaration: `Path` as String

Description
Path of project item. This property is read-only.

ValidateWith

See also

Declaration: `ValidateWith` as String

Description
Used to set the schema/DTD for validation.

XMLForXSLTransformation

See also

Declaration: `XMLForXSLTransformation` as String

Description
Used to set the XML for XSL transformation.

XSLForXMLTransformation

See also

Declaration: `XSLForXMLTransformation` as String

Description
Used to set the XSL for XML transformation.

XSLTransformationFileExtension

See also

Declaration: `XSLTransformationFileExtension` as String

Description
Used to set the file extension for XSL transformation output files.

XSLTransformationFolder

See also

Declaration: `XSLTransformationFolder` as String

Description
Used to set the destination folder for XSL transformation output files.

© 2015 Altova GmbH

3.2.26 SpyProjectItems

See also

Methods
AddFile
AddFolder
AddURL
RemoveItem

Properties
Count
Item

Description
SpyProjectItems Class

AddFile

See also

Declaration: AddFile (*strPath* as String)

Parameters

strPath
Full path with file name of new project item

Description
The method adds a new file to the collection of project items.

AddFolder

See also

Declaration: AddFolder (*strName* as String)

Parameters

strName
Name of the new folder.

Description
The method AddFolder adds a folder with the name strName to the collection of project items.

AddURL

See also

Declaration: AddURL (*strURL* as String, *nURLType* as SPYURLTypes, *strUser* as String, *strPassword* as String, *bSave* as Boolean)

Description

strURL
URL to open as document.

nURLType
Type of document to open. Set to -1 for auto detection.

strUser
Name of the user if required. Can be empty.

strPassword
Password for authentification. Can be empty.

bSave
Save user and password information.

Description
The method adds an URL item to the project collection.

Count

See also

Declaration: Count as long

Description
This property gets the count of project items in the collection. The property is read-only.

Item

See also

Declaration: Item (*n* as long) as SpyProjectItem

Description
Retrieves the n-th element of the collection of project items. The first item has index 1.

RemoveItem

See also

Declaration: RemoveItem (*pItem* as SpyProjectItem)

Description
RemoveItem deletes the item pItem from the collection of project items.

© 2015 Altova GmbH

3.2.27 TextImportExportSettings

See also

Properties for import only
ImportFile

Properties for export only
DestinationFolder
FileExtension
CommentIncluded
RemoveDelimiter
RemoveNewline

Properties for import and export
HeaderRow
FieldDelimiter
EnclosingCharacter
Encoding
EncodingByteOrder

Description
TextImportExportSettings contains options common to text import and export functions.

CommentIncluded

See also

Property: CommentIncluded as Boolean

Description
This property tells whether additional comments are added to the generated text file. Default is true. This property is used only when exporting to text files.

DestinationFolder

See also

Property: DestinationFolder as String

Description
The property DestinationFolder sets the folder where the created files are saved during text export.

EnclosingCharacter

See also

Property: EnclosingCharacter as SPYTextEnclosing

Description
This property defines the character that encloses all field values for import and export. Default is spyNoEnclosing.

Encoding

See also

Property: `Encoding` as String

Description
The property `Encoding` sets the character encoding for the text files for importing and exporting.

EncodingByteOrder

See also

Property: `EncodingByteOrder` as `SPYEncodingByteOrder`

Description
The property `EncodingByteOrder` sets the byte order for Unicode characters. Default is `spyNONE`.

FieldDelimiter

See also

Property: `FieldDelimiter` as `SPYTextDelimiters`

Description
The property `FieldDelimiter` defines the delimiter between the fields during import and export. Default is `spyTabulator`.

FileExtension

See also

Property: `FileExtension` as String

Description
This property sets the file extension for files created on text export.

HeaderRow

See also

Property: `HeaderRow` as Boolean

Description
The property `HeaderRow` is used during import and export. Set `HeaderRow` true on import, if the first line of the text file contains the names of the columns. Set `HeaderRow` true on export, if the first line in the created text files should contain the name of the columns. Default value is true.

© 2015 Altova GmbH

ImportFile

See also

Property: `ImportFile` as String

Description
This property is used to set the text file for import. The string has to be a full qualified path.

RemoveDelimiter

See also

Property: RemoveDelimiter as Boolean

Description
The property RemoveDelimiter defines whether characters in the text that are equal to the delimiter character are removed. Default is false. This property is used only when exporting to text files.

RemoveNewline

See also

Property: RemoveNewline as Boolean

Description
The property RemoveNewline defines whether newline characters in the text are removed. Default is false. This property is used only when exporting to text files.

3.2.28 TextView

See also

Properties and Methods

Application
Parent

LineFromPosition
PositionFromLine
LineLength
SelText
GetRangeText
ReplaceText
MoveCaret
GoToLineChar
SelectText
SelectionStart
SelectionEnd
Text
LineCount
Length

Description

Events

OnBeforeShowSuggestions

See also

Event: OnBeforeShowSuggestions `() as Boolean`

Description
This event gets fired before a suggestion window is shown. The Document property Suggestions contains a string array that is recommended to the user. It is possible to modify the displayed recommendations during this event. Before doing so you have to assign an empty array to the Suggestions property. The best location for this is the OnDocumentOpened event. To prevent the suggestion window to show up return false and true to continue its display.

Examples
Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:

```
Function On_BeforeShowSuggestions()
End Function
```

XMLSpy scripting environment - JScript:

```
function On_BeforeShowSuggestions()
{
}
```


© 2015 Altova GmbH

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (33, ...)  // nEventId = 33
```

OnChar

See also

Event: OnChar(nChar as Long, bExistSuggestion as Boolean) as Boolean

Description

This event gets fired on each key stroke. The parameter nChar is the key that was pressed and bExistSuggestions tells whether a Authentic Desktop generated suggestions window is displayed after this key. The Document property Suggestions contains a string array that is recommended to the user. It is possible to modify the displayed recommendations during this event. Before doing so you have to assign an empty array to the Suggestions property. The best location for this is the OnDocumentOpened event. To prevent the suggestion window to show up return false and true to continue its display.
It is also possible to create a new suggestions window when none is provided by Authentic Desktop. Set the Document property Suggestions to a string array with your recommendations and return true.
This event is fired before the OnBeforeShowSuggestions event. If you prevent to show the suggestion window by returning false then OnBeforeShowSuggestions is not fired.

Examples

Given below are examples of how this event can be scripted.

XMLSpy scripting environment - VBScript:

```
Function On_Char(nChar, bExistSuggestions)
End Function
```

XMLSpy scripting environment - JScript:

```
function On_Char(nChar, bExistSuggestions)
{
}
```

XMLSpy IDE Plugin:

```
IXMLSpyPlugIn.OnEvent (35, ...)  // nEventId = 35
```

Application

Property: Application as Application (read-only)

Description

Access the Authentic Desktop application object.

Errors

3900	The object is no longer valid.
3901	Invalid address for the return parameter was specified.

GetRangeText

Method: GetRangeText(nStart as Long, nEnd as Long) as String

Description
Returns the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

GoToLineChar

Method: GoToLineChar(nLine as Long, nChar as Long)

Description
Moves the caret to the specified line and character position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

Length

Property: Length as Long

Description
Returns the character count of the document.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

LineCount

Property: LineCount as Long

Description
Returns the number of lines in the document.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

LineFromPosition

Method: LineFromPosition(nCharPos as Long) as Long

Description
Returns the line number of the character position.

© 2015 Altova GmbH

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

LineLength

Method: LineLength(nLine as Long) as Long

Description
Returns the length of the line.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

MoveCaret

Method: MoveCaret(nDiff as Long)

Description
Moves the caret nDiff characters.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

Parent

Property: `Parent` as Document (read-only)

Description
Access the parent of the object.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

PositionFromLine

Method: PositionFromLine(nLine as Long) as Long

Description
Returns the start position of the line.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

ReplaceText

Method: ReplaceText(nPosFrom as Long, nPosTill as Long, sText as String)

Description

Replaces the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

SelectionEnd

Property: SelectionEnd as Long

Description
Returns/sets the text selection end position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

SelectionStart

Property: SelectionStart as Long

Description
Returns/sets the text selection start position.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

SelectText

Method: SelectText(nPosFrom as Long, nPosTill as Long)

Description
Selects the text in the specified range.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

SelText

Property: SelText as String

Description
Returns/sets the selected text.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

© 2015 Altova GmbH

Text

Property: Text as String

Description
Returns/sets the document text.

Errors
3900 The object is no longer valid.
3901 Invalid address for the return parameter was specified.

3.2.29 XMLData

See also

Properties
Kind
Name
TextValue

HasChildren
MayHaveChildren
Parent

Methods
GetFirstChild
GetNextChild
GetCurrentChild

InsertChild
InsertChildAfter
InsertChildBefore
AppendChild

EraseAllChildren
EraseChild
EraseCurrentChild

IsSameNode

CountChildren
CountChildrenKind

GetChild
GetChildAttribute
GetChildElement
GetChildKind
GetNamespacePrefixForURI

HasChildrenKind
SetTextValueXMLEncoded

Description
The XMLData interface provides direct XML-level access to a document. You can read and directly modify the XML representation of the document. However, please, note the following restrictions:

- The XMLData representation is only valid when the document is shown in grid view or authentic view.
- When in authentic view, additional XMLData elements are automatically inserted as parents of each visible document element. Typically this is an XMLData of kind `spyXMLDataElement` with the Name property set to 'Text'.
- When you use the XMLData interface while in a different view mode you will not receive errors, but changes are not reflected to the view and might get lost during the next view switch.

© 2015 Altova GmbH

Note also:

- Setting a new text value for an XML element is possible if the element does not have non-text children. A text value can be set even if the element has attributes.
- When setting a new text value for an XML element which has more than one text child, the latter will be deleted and replaced by one new text child.
- When reading the text value of an XML element which has more than one text child, only the value of the first text child will be returned.

AppendChild

See also

Declaration: `AppendChild` (*`pNewData`* as `XMLData`)

Description
`AppendChild` appends `pNewData` as last child to the `XMLData` object.

Errors

1500	The XMLData object is no longer valid.
1505	Invalid XMLData kind was specified.
1506	Invalid address for the return parameter was specified.
1507	Element cannot have Children
1512	Cyclic insertion - new data element is already part of document
1514	Invalid XMLData kind was specified for this position.
1900	Document must not be modified

Example

```
Dim objCurrentParent As XMLData
Dim objNewChild As XMLData

Set objNewChild = objSpy.ActiveDocument.CreateChild(spyXMLDataElement)
Set objCurrentParent = objSpy.ActiveDocument.RootElement

objCurrentParent.AppendChild objNewChild

Set objNewChild = Nothing
```

CountChildren

See also

Declaration: `CountChildren` as long

Description
`CountChildren` gets the number of children.

Available with TypeLibrary version 1.5

Errors

1500	The XMLData object is no longer valid.

CountChildrenKind

See also

Declaration: `CountChildrenKind` (*nKind* as `SPYXMLDataKind`) as long

Description
`CountChildrenKind` gets the number of children of the specific kind.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.

EraseAllChildren

See also

Declaration: `EraseAllChildren`

Description
`EraseAllChildren` deletes all associated children of the `XMLData` object.

Errors
1500 The XMLData object is no longer valid.
1900 Document must not be modified

Example
The sample erases all elements of the active document.

```
Dim objCurrentParent As XMLData

Set objCurrentParent = objSpy.ActiveDocument.RootElement
objCurrentParent.EraseAllChildren
```

EraseChild

Method: `EraseChild` (Child as XMLData)

Description
Deletes the given child node.

Errors
1500 Invalid object.
1506 Invalid input xml
1510 Invalid parameter.

© 2015 Altova GmbH

EraseCurrentChild

See also

Declaration: `EraseCurrentChild`

Description

`EraseCurrentChild` deletes the current `XMLData` child object. Before you call `EraseCurrentChild` you must initialize an internal iterator with `XMLData.GetFirstChild`. After deleting the current child, `EraseCurrentChild` increments the internal iterator of the XMLData element. No error is returned when the last child gets erased and the iterator is moved past the end of the child list. The next call to `EraseCurrentChild` however, will return error 1503.

Errors

1500	The XMLData object is no longer valid.
1503	No iterator is initialized for this XMLData object, or the iterator points past the last child.
1900	Document must not be modified

Examples

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// erase all children of XMLData
// ---------------------------------------
// let's get an XMLData element, we assume that the
// cursor selects the parent of a list in grid view
var objList = Application.ActiveDocument.GridView.CurrentFocus;

// the following line would be shorter, of course
//objList.EraseAllChildren ();

// but we want to demonstrate the usage of EraseCurrentChild
if ((objList != null) && (objList.HasChildren))
{
 try
 {
  objEle = objList.GetFirstChild(-1);
  while (objEle != null)
   objList.EraseCurrentChild();
   // no need to call GetNextChild
 }
 catch (err)
  // 1503 - we reached end of child list
  { if ((err.number & 0xffff) != 1503) throw (err); }
}
```

GetChild

See also

Declaration: `GetChild` (*position* as long) as `XMLData`

Return Value
Returns an XML element as `XMLData` object.

Description
`GetChild()` returns a reference to the child at the given index (zero-based).

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

GetChildAttribute

Method: `GetChildAttribute` (strName as string) child as XMLData object (`NULL` on error)

Description
Retrieves the attribute having the given name.

Errors
1500 Invalid object.
1510 Invalid parameter.

GetChildElement

Method: `GetChildElement` (strName as string, nIndex as long) child as XMLData object (`NULL` on error)

Description
Retrieves the Nth child element with the given name.

Errors
1500 Invalid object.
1510 Invalid parameter.

GetChildKind

See also

Declaration: `GetChildKind` (*`position`* as long, *`nKind`* as `SPYXMLDataKind`) as `XMLData`

Return Value
Returns an XML element as `XMLData` object.

Description
`GetChildKind()` returns a reference to a child of this kind at the given index (zero-based). The

© 2015 Altova GmbH

position parameter is relative to the number of children of the specified kind and not to all children of the object.

Available with TypeLibrary version 1.5

Errors

1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

GetCurrentChild

See also

Declaration: `GetCurrentChild` as `XMLData`

Return Value

Returns an XML element as `XMLData` object.

Description

`GetCurrentChild` gets the current child. Before you call `GetCurrentChild` you must initialize an internal iterator with `XMLData.GetFirstChild`.

Errors

1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1510 Invalid address for the return parameter was specified.

GetFirstChild

See also

Declaration: `GetFirstChild` (*`nKind`* as `SPYXMLDataKind`) as `XMLData`

Return Value

Returns an XML element as `XMLData` object.

Description

`GetFirstChild` initializes a new iterator and returns the first child. Set `nKind` = -1 to get an iterator for all kinds of children.
REMARK: The iterator is stored inside the XMLData object and gets destroyed when the XMLData object gets destroyed. Be sure to keep a reference to this object as long as you want to use `GetCurrentChild`, `GetNextChild` or `EraseCurrentChild`.

Errors

1500 The XMLData object is no longer valid.
1501 Invalid XMLData kind was specified.
1504 Element has no children of specified kind.
1510 Invalid address for the return parameter was specified.

Example

See the example at XMLData.GetNextChild.

GetNamespacePrefixForURI

Method: GetNamespacePrefixForURI (strURI as string) strNS as string

Description
Returns the namespace prefix of the supplied URI.

Errors
1500 Invalid object.
1510 Invalid parameter.

GetNextChild

See also

Declaration: GetNextChild as XMLData

Return Value
Returns an XML element as XMLData object.

Description
GetNextChild steps to the next child of this element. Before you call GetNextChild you must initialize an internal iterator with XMLData.GetFirstChild.

Check for the last child of the element as shown in the sample below.

Errors
1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1510 Invalid address for the return parameter was specified.

Examples

```
' ------------------------------------------------
' VBA code snippet - iterate XMLData children
' ------------------------------------------------
On Error Resume Next
Set objParent = objSpy.ActiveDocument.RootElement

'get elements of all kinds
Set objCurrentChild = objParent.GetFirstChild(-1)

Do
 'do something useful with the child

 'step to next child
 Set objCurrentChild = objParent.GetNextChild
Loop Until (Err.Number - vbObjectError = 1503)
```


© 2015 Altova GmbH

```
// ---------------------------------------
// XMLSpy scripting environment - JScript
// iterate through children of XMLData
// ---------------------------------------
try
{
   var objXMLData = ... // initialize somehow
   var objChild = objXMLData.GetFirstChild(-1);

   while (true)
   {
      // do something usefull with objChild

      objChild = objXMLData.GetNextChild();
   }
}
catch (err)
{
   if ((err.number & 0xffff) == 1504)
    ; // element has no children
   else if ((err.number & 0xffff) == 1503)
    ;// last child reached
   else
    throw (err);
}
```

GetTextValueXMLDecoded

Method: `GetTextValueXMLDecoded ()` as string

Description
Gets the decoded text value of the XML.

Errors
1500 Invalid object.
1510 Invalid parameter.

HasChildren

See also

Declaration: `HasChildren` as Boolean

Description
The property is true if the object is the parent of other `XMLData` objects. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

HasChildrenKind

See also

Declaration: `HasChildrenKind` (*`nKind`* as `SPYXMLDataKind`) as Boolean

Description
The method returns true if the object is the parent of other `XMLData` objects of the specific kind.

Available with TypeLibrary version 1.5

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

InsertChild

See also

Declaration: `InsertChild` (*`pNewData`* as `XMLData`)

Description
`InsertChild` inserts the new child before the current child (see also `XMLData.GetFirstChild`, `XMLData.GetNextChild` to set the current child).

Errors
1500 The XMLData object is no longer valid.
1503 No iterator is initialized for this XMLData object.
1505 Invalid XMLData kind was specified.
1506 Invalid address for the return parameter was specified.
1507 Element cannot have Children
1512 Cyclic insertion - new data element is already part of document
1514 Invalid XMLData kind was specified for this position.
1900 Document must not be modified

InsertChildAfter

Method: `InsertChildBefore` (Node as XMLData, NewData as XMLData)

Description
Inserts a new XML node (supplied with the second parameter) after the specified node (first parameter).

Errors
1500 Invalid object.
1506 Invalid input xml
1507 No children allowed
1510 Invalid parameter.
1512 Child is already added
1514 Invalid kind at position

© 2015 Altova GmbH

InsertChildBefore

Method: `InsertChildBefore` (Node as XMLData, NewData as XMLData)

Description
Inserts a new XML node (supplied with the second parameter) before the specified node (first parameter).

Errors
1500 Invalid object.
1506 Invalid input xml
1507 No children allowed
1510 Invalid parameter.
1512 Child is already added
1514 Invalid kind at position

IsSameNode

See also

Declaration: `IsSameNode` (*`pNodeToCompare`* as `XMLData`) as Boolean

Description

Returns true if `pNodeToCompare` references the same node as the object itself.

Errors
1500 The XMLData object is no longer valid.
1506 Invalid address for the return parameter was specified.

Kind

See also

Declaration: `Kind` as `SPYXMLDataKind`

Description
Kind of this `XMLData` object. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

MayHaveChildren

See also

Declaration: `MayHaveChildren` as Boolean

Description
Indicates whether it is allowed to add children to this `XMLData` object.
This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

Name

See also

Declaration: `Name` as String

Description
Used to modify and to get the name of the `XMLData` object.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

Parent

See also

Declaration: `Parent` as `XMLData`

Return value
Parent as `XMLData` object. Nothing (or NULL) if there is no parent element.

Description
Parent of this element. This property is read-only.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

SetTextValueXMLEncoded

Method: `SetTextValueXMLEncoded` (*`strVal`* as `String`)

Description
Sets the encoded text value of the XML.

Errors
1500 Invalid object.
1513 Modification not allowed.

© 2015 Altova GmbH

TextValue

See also

Declaration: `TextValue` as String

Description
Used to modify and to get the text value of this `XMLData` object.

Errors
1500 The XMLData object is no longer valid.
1510 Invalid address for the return parameter was specified.

3.3 Enumerations

This is a list of all enumerations used by the Authentic Desktop API. If your scripting environment does not support enumerations use the number-values instead.

© 2015 Altova GmbH

3.3.1 ENUMApplicationStatus

Description
Enumeration to specify the current Application status.

Possible values:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3
eApplicationProcessingCommandLine	= 4

3.3.2 SPYAttributeTypeDefinition

Description

Attribute type definition that can be selected for generation of Sample XML.
This type is used with the method GenerateDTDOrSchema and GenerateDTDOrSchemaEx.

Possible values:

spyMergedGlobal	= 0
spyDistinctGlobal	= 1
spyLocal	= 2

© 2015 Altova GmbH

3.3.3 SPYAuthenticActions

Description
Actions that can be performed on AuthenticRange objects.

Possible values:

spyAuthenticInsertAt	= 0
spyAuthenticApply	= 1
spyAuthenticClearSurr	= 2
spyAuthenticAppend	= 3
spyAuthenticInsertBefore	= 4
spyAuthenticRemove	= 5

3.3.4 SPYAuthenticDocumentPosition

Description
Relative and absolute positions used for navigating with AuthenticRange objects.

Possible values:

spyAuthenticDocumentBegin = 0
spyAuthenticDocumentEnd = 1
spyAuthenticRangeBegin = 2
spyAuthenticRangeEnd = 3

© 2015 Altova GmbH

3.3.5 SPYAuthenticElementActions

Description

Actions that can be used with the obsolete object GetAllowedElements (superseded by `AuthenticRange.CanPerformActionWith`).

Possible values:

k_ActionInsertAt	= 0
k_ActionApply	= 1
k_ActionClearSurr	= 2
k_ActionAppend	= 3
k_ActionInsertBefore	= 4
k_ActionRemove	= 5

3.3.6 SPYAuthenticElementKind

Description

Enumeration of the different kinds of elements used for navigation and selection within the AuthenticRange and AuthenticView objects.

Possible values:

spyAuthenticChar	= 0
spyAuthenticWord	= 1
spyAuthenticLine	= 3
spyAuthenticParagraph	= 4
spyAuthenticTag	= 6
spyAuthenticDocument	= 8
spyAuthenticTable	= 9
spyAuthenticTableRow	= 10
spyAuthenticTableColumn	= 11

© 2015 Altova GmbH

3.3.7 SPYAuthenticMarkupVisibility

Description
Enumeration values to customize the visibility of markup with MarkupVisibility.

Possible values:

spyAuthenticMarkupHidden	= 0
spyAuthenticMarkupSmall	= 1
spyAuthenticMarkupLarge	= 2
spyAuthenticMarkupMixed	= 3

3.3.8 SPYAuthenticToolbarButtonState

Description
Authentic toolbar button states are given by the following enumeration:

Possible values:

`authenticToolbarButtonDefault`	= 0
`authenticToolbarButtonEnabled`	= 1
`authenticToolbarButtonDisabled`	= 2

© 2015 Altova GmbH

3.3.9 SPYDatabaseKind

Description

Values to select different kinds of databases for import. See `DatabaseConnection.DatabaseKind` for its use.

Possible values:

spyDB_Access	= 0
spyDB_SQLServer	= 1
spyDB_Oracle	= 2
spyDB_Sybase	= 3
spyDB_MySQL	= 4
spyDB_DB2	= 5
spyDB_Other	= 6
spyDB_Unspecified	= 7
spyDB_PostgreSQL	= 8
spyDB_iSeries	= 9

3.3.10 SPYDialogAction

Description
Values to simulate different interactions on dialogs. See Dialogs for all dialogs available.

Possible values:

```
spyDialogOK          = 0    // simulate click on OK button
spyDialogCancel      = 1    // simulate click on Cancel button
spyDialogUserInput   = 2    // show dialog and allow user
                            interaction
```


© 2015 Altova GmbH

3.3.11 SPYDOMType

Description
Enumeration values to parameterize generation of C++ code from schema definitions.

Possible values:

spyDOMType_msxml4	= 0	Obsolete
spyDOMType_xerces	= 1	
spyDOMType_xerces3	= 2	
spyDOMType_msxml6	= 3	

`spyDOMType_xerces` indicates Xerces 2.x usage; `spyDOMType_xerces3` indicates Xerces 3.x usage.

3.3.12 SPYDTDSchemaFormat

Description
Enumeration to identify the different schema formats.

Possible values:

spyDTD	= 0
spyW3C	= 1

© 2015 Altova GmbH

3.3.13 SPYEncodingByteOrder

Description
Enumeration values to specify encoding byte ordering for text import and export.

Possible values:

spyNONE	= 0
spyLITTLE_ENDIAN	= 1
spyBIG_ENDIAN	= 2

3.3.14 SPYExportNamespace

Description
Enumeration type to configure handling of namespace identifiers during export.

Possible values:

spyNoNamespace	= 0
spyReplaceColonWithUnderscore	= 1

© 2015 Altova GmbH

3.3.15 SPYFindInFilesSearchLocation

Description
The different locations where a search can be performed. This type is used with the FindInFilesDlg dialog.

Possible values:

spyFindInFiles_Documents	= 0
spyFindInFiles_Project	= 1
spyFindInFiles_Folder	= 2

3.3.16 SPYFrequentElements

Description
Enumeration value to parameterize schema generation.

Possible values:

spyGlobalElements	= 0
spyGlobalComplexType	= 1

© 2015 Altova GmbH

3.3.17 SPYImageKind

Description

Enumeration values to parameterize image type of the generated documentation. These values are used in SchemaDocumentationDialog.DiagramFormat.

Possible values:

spyImageType_PNG	= 0
spyImageType_EMF	= 1

3.3.18 SPYImportColumnsType

Description
Enumeration to specify different Import columns types.

Possible values:

spyImportColumns_Element	= 0
spyImportColumns_Attribute	= 1

© 2015 Altova GmbH

3.3.19 SPYKeyEvent

Description

Enumeration type to identify the different key events. These events correspond with the equally named windows messages.

Possible values:

spyKeyDown	= 0
spyKeyUp	= 1
spyKeyPressed	= 2

3.3.20 SPYKeyStatus

Description
Enumeration type to identify the key status.

Possible values:

spyLeftShiftKeyMask	= 1
spyRightShiftKeyMask	= 2
spyLeftCtrlKeyMask	= 4
spyRightCtrlKeyMask	= 8
spyLeftAltKeyMask	= 16
spyRightAltKeyMask	= 32

© 2015 Altova GmbH

3.3.21 SPYLibType

Description
Enumeration values to parameterize generation of C++ code from schema definitions.

Possible values:

spyLibType_static	= 0
spyLibType_dll	= 1

3.3.22 SPYLoading

Description
Enumeration values to define loading behaviour of URL files.

Possible values:

spyUseCacheProxy	= 0
spyReload	= 1

© 2015 Altova GmbH

3.3.23 SPYMouseEvent

Description

Enumeration type that defines the mouse status during a mouse event. Use the enumeration values as bitmasks rather then directly comparing with them.

Examples

```
' to check for ctrl-leftbutton-down in VB
If (i_eMouseEvent = (XMLSpyLib.spyLeftButtonDownMask Or
XMLSpyLib.spyCtrlKeyDownMask)) Then
       ' react on ctrl-leftbutton-down
End If

' to check for double-click with any button in VBScript
If (((i_eMouseEvent And spyDoubleClickMask) <> 0) Then
       ' react on double-click
End If
```

Possible values:

spyNoButtonMask	= 0	
spyMouseMoveMask	= 1	
spyLeftButtonMask	= 2	
spyMiddleButtonMask	= 4	
spyRightButtonMask	= 8	
spyButtonUpMask	= 16	
spyButtonDownMask	= 32	
spyDoubleClickMask	= 64	
spyShiftKeyDownMask	= 128	
spyCtrlKeyDownMask	= 256	
spyLeftButtonDownMask	= 34	// spyLeftButtonMask | spyButtonDownMask
spyMiddleButtonDownMask	= 36	// spyMiddleButtonMask | spyButtonDownMask
spyRightButtonDownMask	= 40	// spyRightButtonMask | spyButtonDownMask
spyLeftButtonUpMask	= 18	// spyLeftButtonMask | spyButtonUpMask
spyMiddleButtonUpMask	= 20	// spyMiddleButtonMask | spyButtonUpMask
spyRightButtonUpMask	= 24	// spyRightButtonMask | spyButtonUpMask
spyLeftDoubleClickMask	= 66	// spyRightButtonMask | spyButtonUpMask
spyMiddleDoubleClickMask	= 68	// spyMiddleButtonMask | spyDoubleClickMask
spyRightDoubleClickMask	= 72	// spyRightButtonMask | spyDoubleClickMask

3.3.24 SPYNumberDateTimeFormat

Description
Enumeration value to configure database connections.

Possible values:

spySystemLocale	= 0
spySchemaCompatible	= 1

© 2015 Altova GmbH

3.3.25 SPYProgrammingLanguage

Description
Enumeration values to select the programming language for code generation from schema definitions.

Only available/enabled in the Enterprise edition. An error is returned, if accessed by any other version.

Possible values:

spyUndefinedLanguage	= -1
spyJava	= 0
spyCpp	= 1
spyCSharp	= 2

3.3.26 SPYProjectItemTypes

Description

Enumeration values to identify the different elements in project item lists. See SpyProjectItem.ItemType.

Possible values:

spyUnknownItem	= 0
spyFileItem	= 1
spyFolderItem	= 2
spyURLItem	= 3

© 2015 Altova GmbH

3.3.27 SPYProjectType

Description

Enumeration values to parameterize generation of C# from schema definitions.

Possible values:

spyVisualStudioProject	= 0	Obsolete
spyVisualStudio2003Project	= 1	Obsolete
spyBorlandProject	= 2	Obsolete
spyMonoMakefile	= 3	Obsolete
spyVisualStudio2005Project	= 4	For C++ code also
spyVisualStudio2008Project	= 5	For C++ code also
spyVisualStudio2010Project	= 6	For C++ code also

3.3.28 SpySampleXMLGenerationChoiceMode

Description

This enumeration is used in GenerateSampleXMLDlg.ChoiceMode:

spySampleXMLGen_FirstBranch = 0
spySampleXMLGen_AllBranches = 1
spySampleXMLGen_ShortestBranch = 2

© 2015 Altova GmbH

3.3.29 SPYSampleXMLGenerationOptimization (Obsolete)

This enumeration is OBSOLETE since v2014.

Description
Specify the elements that will be generated in the Sample XML.
This enumeration is used in GenerateSampleXMLDlg.

Possible values:

spySampleXMLGen_Optimized	= 0
spySampleXMLGen_NonMandatoryElements	= 1
spySampleXMLGen_Everything	= 2

3.3.30 SpySampleXMLGenerationSampleValueHints

Description
This enumeration is used in GenerateSampleXMLDlg.SampleValueHints

spySampleXMLGen_FirstFit = 0
spySampleXMLGen_RandomFit = 1
spySampleXMLGen_CycleThrough = 2

© 2015 Altova GmbH

3.3.31 SPYSampleXMLGenerationSchemaOrDTDAssignment

Description
Specifies what kind of reference to the schema/DTD should be added to the generated Sample XML.
This enumeration is used in GenerateSampleXMLDlg.

Possible values:

spySampleXMLGen_AssignRelatively	= 0
spySampleXMLGen_AssignAbsolutely	= 1
spySampleXMLGen_DoNotAssign	= 2

3.3.32 SPYSchemaDefKind

Description
Enumeration type to select schema diagram types.

Possible values:

spyKindElement	= 0
spyKindComplexType	= 1
spyKindSimpleType	= 2
spyKindGroup	= 3
spyKindModel	= 4
spyKindAny	= 5
spyKindAttr	= 6
spyKindAttrGroup	= 7
spyKindAttrAny	= 8
spyKindIdentityUnique	= 9
spyKindIdentityKey	= 10
spyKindIdentityKeyRef	= 11
spyKindIdentitySelector	= 12
spyKindIdentityField	= 13
spyKindNotation	= 14
spyKindInclude	= 15
spyKindImport	= 16
spyKindRedefine	= 17
spyKindFacet	= 18
spyKindSchema	= 19
spyKindCount	= 20

© 2015 Altova GmbH

3.3.33 SPYSchemaDocumentationFormat

Description

Enumeration values to parameterize generation of schema documentation. These values are used in `SchemaDocumentationDialog.OutputFormat`.

Possible values:

spySchemaDoc_HTML	= 0
spySchemaDoc_MSWord	= 1
spySchemaDoc_RTF	= 2
spySchemaDoc_PDF	= 3

3.3.34 SPYSchemaExtensionType

Description
Enumeration to specify different Schema Extension types.

Possible values:

spySchemaExtension_None	= 0
spySchemaExtension_SQL_XML	= 1
spySchemaExtension_MS_SQL_Server	= 2
spySchemaExtension_Oracle	= 3

© 2015 Altova GmbH

3.3.35 SPYSchemaFormat

Description
Enumeration to specify different Schema Format types.

Possible values:

spySchemaFormat_Hierarchical	= 0
spySchemaFormat_Flat	= 1

3.3.36 SPYTextDelimiters

Description
Enumeration values to specify text delimiters for text export.

Possible values:

spyTabulator	= 0
spySemicolon	= 1
spyComma	= 2
spySpace	= 3

© 2015 Altova GmbH

3.3.37 SPYTextEnclosing

Description
Enumeration value to specify text enclosing characters for text import and export.

Possible values:

spyNoEnclosing	= 0
spySingleQuote	= 1
spyDoubleQuote	= 2

3.3.38 SPYTypeDetection

Description

Enumeration to select how type detection works during GenerateDTDOrSchema and GenerateDTDOrSchemaEx.

Possible values:

spyBestPossible = 0
spyNumbersOnly = 1
spyNoDetection = 2

© 2015 Altova GmbH

3.3.39 SPYURLTypes

Description
Enumeration to specify different URL types.

Possible values:
spyURLTypeAuto = -1
spyURLTypeXML = 0
spyURLTypeDTD = 1

3.3.40 SPYValidateXSDVersion

Description

Enumeration values that select what XSD version to use. The XSD version that is selected depends on both (i) the presence/absence—and, if present, the value—of the `/xs:schema/@vc:minVersion` attribute of the XSD document, and (ii) the value of this enumeration.

`spyValidateXSDVersion_1_0` selects `XSD 1.0` if `vc:minVersion` is absent, or is present with any value.
`spyValidateXSDVersion_1_1` selects `XSD 1.1` if `vc:minVersion` is absent, or is present with any value.
`spyValidateXSDVersion_AutoDetect` selects `XSD 1.1` if `vc:minVersion=1.1`. If the `vc:minVersion` attribute is absent, or is present with a value other than `1.1`, then `XSD 1.0` is selected.

Possible values

```
spyValidateXSDVersion_AutoDetect       = 0
spyValidateXSDVersion_1_1              = 1
spyValidateXSDVersion_1_0              = 2
```


© 2015 Altova GmbH

3.3.41 SPYValidateErrorFormat

Description
Enumeration values that select the format of the error message.

Possible values

```
spyValidateErrorFormat_Text          = 0
spyValidateErrorFormat_ShortXML      = 1
spyValidateErrorFormat_LongXML       = 2
```

3.3.42 SPYViewModes

Description

Enumeration values that define the different view modes for XML documents. The mode *spyViewAuthentic(4)* identifies the mode that was intermediately called DocEdit mode and is now called Authentic mode. The mode *spyViewJsonSchema* identifies a mode which is mapped to the Schema Design View on the GUI but is distinguished internally.

Possible values:

spyViewGrid	= 0	
spyViewText	= 1	
spyViewBrowser	= 2	
spyViewSchema	= 3	
spyViewContent	= 4	// obsolete
spyViewAuthentic	= 4	
spyViewWSDL	= 5	
spyViewZIP	= 6	
spyViewEditionInfo	= 7	
spyViewXBRL	= 8	
spyViewJsonSchema	= 9	

© 2015 Altova GmbH

3.3.43 SPYVirtualKeyMask

Description

Enumeration type for the most frequently used key masks that identify the status of the virtual keys. Use these values as bitmasks rather then directly comparing with them. When necessary, you can create further masks by using the 'logical or' operator.

Examples

```
'  VBScript sample: check if ctrl-key is pressed
If ((i_nVirtualKeyStatus And spyCtrlKeyMask) <> 0)) Then
       ' ctrl-key is pressed
End If

'  VBScript sample: check if ONLY ctrl-key is pressed
If (i_nVirtualKeyStatus == spyCtrlKeyMask) Then
       ' exactly ctrl-key is pressed
End If

// JScript sample: check if any of the right virtual keys is pressed
if ((i_nVirtualKeyStatus & (spyRightShiftKeyMask | spyRightCtrlKeyMask |
spyRightAltKeyMask)) != 0)
{
       ; ' right virtual key is pressed
}
```

Possible values:

spyNoVirtualKeyMask	= 0	
spyLeftShiftKeyMask	= 1	
spyRightShiftKeyMask	= 2	
spyLeftCtrlKeyMask	= 4	
spyRightCtrlKeyMask	= 8	
spyLeftAltKeyMask	= 16	
spyRightAltKeyMask	= 32	
spyShiftKeyMask	= 3	// spyLeftShiftKeyMask | spyRightShiftKeyMask
spyCtrlKeyMask	= 12	// spyLeftCtrlKeyMask | spyRightCtrlKeyMask
spyAltKeyMask	= 48	// spyLeftAltKeyMask | spyRightAltKeyMask

3.3.44 SPYXMLDataKind

Description
The different types of XMLData elements available for XML documents.

Possible values:

spyXMLDataXMLDocStruct	= 0
spyXMLDataXMLEntityDocStruct	= 1
spyXMLDataDTDDocStruct	= 2
spyXMLDataXML	= 3
spyXMLDataElement	= 4
spyXMLDataAttr	= 5
spyXMLDataText	= 6
spyXMLDataCData	= 7
spyXMLDataComment	= 8
spyXMLDataPI	= 9
spyXMLDataDefDoctype	= 10
spyXMLDataDefExternalID	= 11
spyXMLDataDefElement	= 12
spyXMLDataDefAttlist	= 13
spyXMLDataDefEntity	= 14
spyXMLDataDefNotation	= 15
spyXMLDataKindsCount	= 16

© 2015 Altova GmbH

4 ActiveX Integration

AuthenticDesktopControl is a control that provides a means of integration of the Authentic Desktop user interface and the functionality described in this section into most kinds of applications. ActiveX technology was chosen so as to allow integration using any of a wide variety of languages; this enables C++, C#, VisualBasic, or HTML to be used for integration (ActiveX components integrated in HTML officially only work with Microsoft Internet Explorer). The attached Java wrapper library allows integration into Java. All components are full OLE Controls, which makes integration as simple as possible. Two different levels of integration are provided, thus enabling the integration to be adapted to a wide range of needs.

To integrate Authentic Desktop you must install the Authentic Desktop Integration Package. Ensure that you install Authentic Desktop first, and then the Authentic Desktop Integration Package. For the integration pack to work correctly, JRE 6 (or later) must be installed.

For a successful integration you have to consider the following main design factors:

- What technology or programming language can the hosting application use to integrate the AuthenticDesktopControl?
- Should the integrated UI look exactly like Authentic Desktop with all its menus, toolbars, and windows, or will a subset of these elements—like allowing only one document and a restricted set of commands—be more effective?
- How deep will the integration be? Should the Authentic Desktop user interface be used as is? Are user interface extensions and/or restrictions required? Can some frequently used tasks be automated?

The sections, Integration at the Application Level and Integration at Document Level describe the key steps at these respective levels. The Programming Languages section provides examples in C#, HTML, and Java. Looking through these examples will help you to make the right decisions quickly. The section, Object Reference, describes all COM objects that can be used for integration, together with their properties and methods.

For automation tasks, the Authentic Desktop Automation Interface is accessible from the AuthenticDesktopControl as well.

For information about how to integrate Authentic Desktop into Microsoft Visual Studio see the section, Authentic Desktop in Visual Studio.

4.1 Integration at Application Level

Integration at application level is simple and straightforward. It allows you to embed the complete interface of Authentic Desktop into a window of your application. Since you get the whole user interface of Authentic Desktop, you get all menus, toolbars, the status bar, document windows, and helper windows. Customization of the application's user interface is restricted to what Authentic Desktop provides. This includes rearrangement and resizing of helper windows and customization of menus and toolbars.

The only ActiveX control you need to integrate is `AuthenticDesktopControl`. Its property `IntegrationLevel` defaults to application-level. You may use `Appearance` and `BorderStyle` to configure the appearance of the control's wrapper window. Do not instantiate or access `AuthenticDesktopControlDocument` or `AuthenticDesktopControlPlaceHolder` ActiveX controls when integrating at application-level.

If you have any initialization to do or if you want to automate some behaviour of Authentic Desktop, use the properties, methods, and events described for `AuthenticDesktopControl`. Consider using `AuthenticDesktopControl.Application` for more complex access to Authentic Desktop functionality.

In the Programming Languages | HTML section is an example (Integration at Application Level) that shows how the Authentic Desktop application can be embedded in an HTML page. For usage with other programming languages, or more sophisticated access, see the Examples of integration at document-level (in C#, HTML, and Java).

© 2015 Altova GmbH

4.2 Integration at Document Level

Integration at document level gives you freedom over instantiation and placement of the following parts of the Authentic Desktop user interface:

If necessary, a replacement for the menus and toolbars of Authentic Desktop must be provided by your application.

You will need to instantiate and access multiple ActiveX controls, depending on which user interface parts you want to re-use. All these controls are contained in the AuthenticDesktopControl OCX.

- Use AuthenticDesktopControl to set the integration level and access application wide functionality.
- Use AuthenticDesktopControlDocument to create any number of editor windows. It may be sufficient to create only one window and re-use it, depending on your needs.
- Optionally Use AuthenticDesktopControlPlaceholder to embed Authentic Desktop entry helper windows, validator output or other windows mentioned above.
- Access run-time information about commands, menus, and toolbars available in AuthenticDesktopControl to seamlessly integrate these commands into your application's menus and toolbars. See Query Authentic Desktop Commands for more information.

If you want to automate some behaviour of Authentic Desktop use the properties, methods, and events described for the AuthenticDesktopControl, AuthenticDesktopControlDocument and AuthenticDesktopControlPlaceHolder. Consider using AuthenticDesktopControl.Application, AuthenticDesktopControlDocument.Document and AuthenticDesktopControlPlaceHolder.Project for more complex access to Authentic Desktop functionality. However, to open a document always use AuthenticDesktopControlDocument.Open or AuthenticDesktopControlDocument.New on the appropriate document control. To open a project always use AuthenticDesktopControlPlaceHolder.OpenProject on a placeholder control embedding a Authentic Desktop project window.

See Examples on how to instantiate and access the necessary controls in different programming environments.

4.2.1 Use AuthenticDesktopControl

To integrate at document level, instantiate a `AuthenticDesktopControl` first. Set the property `IntegrationLevel` to `ICActiveXIntegrationOnDocumentLevel (= 1)`. Set the window size of the embedding window to `0x0` to hide any user interface behind the control. You may use `Appearance` and `BorderStyle` to configure the appearance of the control's wrapper window.

Avoid using the method `Open` since this might lead to unexpected results. Use the corresponding open methods of `AuthenticDesktopControlDocument` and `AuthenticDesktopControlPlaceHolder`, instead.

See Query Authentic Desktop Commands for a description of how to integrate Authentic Desktop commands into your application. Send commands to Authentic Desktop via the method `Exec`. Query if a command is currently enabled or disabled using the method `QueryStatus`.

© 2015 Altova GmbH

4.2.2 Use AuthenticDesktopControlDocument

An instance of the `AuthenticDesktopControlDocument` ActiveX control allows you to embed one Authentic Desktop document editing window into your application. You can use any number of instances you need.

Use the method `Open` to load any other existing file.

The control does not support a read-only mode. The value of the property `ReadOnly` is ignored.

Use `Path` and `Save` or methods and properties accessible via the property `Document` to access document functionality.

4.2.3 Use AuthenticDesktopControlPlaceHolder

Instances of AuthenticDesktopControlPlaceHolder ActiveX controls allow you to selectively embed the additional helper windows of Authentic Desktop into your application. The property `PlaceholderWindowID` selects the Authentic Desktop helper window to be embedded. Use only one AuthenticDesktopControlPlaceHolder for each window identifier. See for valid window identifiers.

For placeholder controls that select the Authentic Desktop project window, additional methods are available. Use `OpenProject` to load a Authentic Desktop project. Use the property `Project` and the methods and properties from the Authentic Desktop automation interface to perform any other project related operations.

© 2015 Altova GmbH

4.2.4 Query Authentic Desktop Commands

When integrating at document-level, no menu or toolbar from Authentic Desktop is available to your application. Instead, you can query all the commands and the structure of the application menu at runtime. Professional applications will need to integrate this menu in a sophisticated manner into their own menu structure. Your installation of Authentic Desktop even provides you with command label images used within Authentic Desktop. See the folder `AuthenticExamples\ActiveX\Images` of your Authentic Desktop installation for icons in GIF format. The file names correspond to the `labels` of commands.

4.3 Programming Languages

This section contains examples of Authentic Desktop document-level integration using different container environments and programming languages. (The HTML section additionally contains examples of integration at application level.) Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX` of your Authentic Desktop installation.

© 2015 Altova GmbH

4.3.1 C#

The C# example shows how to integrate the AuthenticDesktopControl in a common desktop application created with C# using Visual Studio 2008.

Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#` of your Authentic Desktop installation. Please note that the example application is already complete. There is no need to change anything if you want to run it and see it working.

Introduction

Adding the Authentic Desktop components to the Toolbox
Before you take a look at the sample project please add the assemblies to the .NET IDE Toolbox. The Authentic Desktop Installer will have already installed the assemblies in the .NET Global Assembly Cache (GAC). If you open the Toolbox dialog under **Tools | Add/Remove Toolbox Items** the controls will appear as `AxAuthenticDesktopControl`, `AxAuthenticDesktopControlDocument` and `AxAuthenticDesktopControlPlaceholder` on the .NET Framework Components tab. Check all to make them available to the IDE.

Placing the AuthenticDesktopControl

It is necessary to have one AuthenticDesktopControl instance to set the integration level and to manage the Document and Placeholder controls of the Authentic Desktop library. The control is accessible via the General section of the Toolbox helper window in the IDE. To add it you need to select the component in the Toolbox window and drag a rectangle wherever you want to have it in the destination window. If you have an application which does not open a window on startup you can use a simple invisible Form with the control on it which is created manually in the code.

The example project adds this instance to the main MdiContainer MDIMain. If you open MDIMain in the Design View from the Solution Explorer you will see a light blue rectangle at the top-left side in the client area of the Frame window. Selecting this rectangle will show you the properties of the AuthenticDesktopControl. It is important to set the `IntegrationLevel` property to **`ICActiveXIntegrationOnDocumentLevel`** in order to turn on the Document and Placeholder support of the Authentic Desktop library.

4.3.2 HTML

The code listings in this section show how to integrate the AuthenticDesktopControl at application level and document level. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\HTML` of your Authentic Desktop installation.

Integration at Application Level

This example shows a simple integration of the Authentic Desktop control at application-level into a HTML page. The integration is described in the following sections:

- Instantiate a AuthenticDesktopControl in HTML code.
- Implement buttons to load documents and automate code-generation tasks.
- Define actions for some application events.

The code for this example is available at the following location in your Authentic Desktop installation:
`AuthenticExamples\ActiveX\HTML\AuthenticActiveX_ApplicationLevel.htm`.

Note: This example works only in Internet Explorer.

Instantiate the Control

The HTML `Object` tag is used to create an instance of the AuthenticDesktopControl. The `Classid` is that of AuthenticDesktopControl. Width and height specify the window size. No additional parameters are necessary, since application-level is the default.

Add Button to Open Default Document

As a simple example of how to automate some tasks, we add a button to the page:

```
<input type="button" value="Open Marketing Expenses" onclick="BtnOpenMEFile()">
```

When clicked, a predefined document will be opened in the AuthenticDesktopControl. We use a method to locate the file relative to the AuthenticDesktopControl so the example can run on different installations.

Connect to Custom Events

The example implements two event callbacks for AuthenticDesktopControl custom events to show the principle:

```
<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentOpened" of AuthenticDesktopControl object  -->
<SCRIPT LANGUAGE="javascript">
       function objAuthenticDesktopControl::OnDocumentOpened( objDocument )
       {
       // alert("Document '" + objDocument.Name + "' opened!");
       }
</SCRIPT>

<!-- ------------------------------------------------------------ -->
```


© 2015 Altova GmbH

```
<!--  custom event 'OnDocumentClosed" of AuthenticDesktopControl object  -->
<SCRIPT LANGUAGE="javascript">
       function objAuthenticDesktopControl::OnDocumentClosed( objDocument )
       {
       // alert("Document '" + objDocument.Name + "' closed!");
       }
</SCRIPT>
```

Integration at Document Level

This example shows an integration of the Authentic Desktop control at document-level into a HTML page. The following topics are covered:

- Instantiate a AuthenticDesktopControl ActiveX control object in HTML code
- Instantiate a AuthenticDesktopControlDocument ActiveX control to allow editing a Authentic Desktop file
- Instantiate one AuthenticDesktopControlPlaceHolder for a AuthenticDesktopControl project window
- Instantiate one AuthenticDesktopControlPlaceHolder to alternatively host one of the Authentic Desktop helper windows
- Create a simple customer toolbar for some heavy-used Authentic Desktop commands
- Add some more buttons that use the COM automation interface of Authentic Desktop
- Use event handlers to update command buttons

This example is available in its entirety in the file `Authentic DesktopActiveX_ApplicationLevel.htm` within the `C:\Program Files\Altova\Authentic Desktop2016\Examples\ActiveX\HTML\` folder of your Authentic Desktop installation.

Note: This example works only in Internet Explorer.

Instantiate the AuthenticDesktopControl

AuthenticDesktopControlThe HTML `OBJECT` tag is used to create an instance of the AuthenticDesktopControl. The Classid is that of AuthenticDesktopControl. Width and height are set to 0 since we use this control as manager control without use for its user interface. The integration level is specified as a parameter within the `OBJECT` tag.

```
<OBJECT id=""
        width="0"
        height="0"
        VIEWASTEXT>
   <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

Create Editor Window

The HTML `OBJECT` tag is used to embed an editing window.

```
<OBJECT id="objDoc1"
        width="600"
```

```
        height="500"
        VIEWASTEXT>
</OBJECT>
```

Create Project Window

The HTML `OBJECT` tag is used to create a AuthenticDesktopControlPlaceHolder window. The first additional custom parameter defines the placeholder to show the Authentic Desktop project window. The second parameter loads one of the example projects delivered with your Authentic Desktop installation (located in the `<yourusername>/MyDocuments` folder).

```
<OBJECT id="objProjectWindow"
        width="200"
        height="200"
        VIEWASTEXT>
</OBJECT>
```

Create Placeholder for Helper Windows

The HTML `OBJECT` tag is used to instantiate a AuthenticDesktopControlPlaceHolder ActiveX control that can host the different Authentic Desktop helper windows. Initially, no helper window is shown. See the example file.

```
<OBJECT id="objEHWindow"
        width="200"
        height="200"
        VIEWASTEXT>
       <PARAM name="PlaceholderWindowID" value="">
</OBJECT>
```

Three buttons allow us to switch the actual window that will be shown. The JavaScript execute on-button-click sets the property `PlaceHolderWindowID` to the corresponding value defined in

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
//
-------------------------------------------------------------------------------
// specify which of the windows shall be shown in the placeholder control.
function BtnHelperWindow(i_ePlaceholderWindowID)
{
       objEHWindow.PlaceholderWindowID = i_ePlaceholderWindowID;
}
</SCRIPT>
```


© 2015 Altova GmbH

4.3.3 Java

Authentic Desktop ActiveX components can be accessed from Java code. To allow this, the libraries listed below must reside in the classpath. These libraries are partly delivered with the Authentic Desktop Integration Package and are placed in the folder: `JavaAPI` in the Authentic Desktop application folder.

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (in case of the 32-bit installation of Authentic Desktop)
- `AltovaAutomation_x64.dll`: a JNI wrapper for Altova automation servers (in case of the 64-bit installation of Authentic Desktop)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `AuthenticActiveX.jar`: Java classes that wrap the Authentic ActiveX interface
- `AuthenticActiveX_JavaDoc.zip`: a Javadoc file containing help documentation for the Java interface

Note: In order to use the Java ActiveX integration, the DLL and Jar files must be on the Java Classpath.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details of the example Java project, see the section, Example Java Project.

Rules for mapping the ActiveX Control names to Java

The rules for mapping between the ActiveX controls and the Java wrapper are as follows:

- ***Classes and class names***
  For every component of the Authentic Desktop ActiveX interface a Java class exists with the name of the component.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the AuthenticDesktopControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

- ***Enumerations***
  For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `AuthenticDesktopControl`: Java class to access the application

AuthenticDesktopControlEvents: Events interface for the AuthenticDesktopControl
AuthenticDesktopControlEventsDefaultHandler: Default handler for AuthenticDesktopControlEvents

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Java name
AuthenticDesktopControlDocument, method New	newDocument
Document, method SetEncoding	setFileEncoding
AuthenticView, method Goto	gotoElement
AuthenticRange, method Goto	gotoElement
AuthenticRange, method Clone	cloneRange

The XMLSpyActiveX.jar library

The classes and interfaces contained in the jar file are part of the com.altova.automation.XMLSpy package. They are described in the XMLSpyActiveX_JavaDoc.zip file from the folder JavaAPI in the Authentic Desktop application folder.

This section

This section shows how some basic Authentic Desktop ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project
- Creating the ActiveX Controls
- Loading Data in the Controls
- Basic Event Handling
- Menus
- UI Update Event Handling
- Creating an XML Tree

Example Java Project

The Authentic Desktop installation package contains an example Java project, located in the ActiveX Examples folder of the application folder: <ApplicationFolder>\Examples\ActiveX\Java\.

The Java example shows how to integrate the AuthenticDesktopControl in a common desktop application created with Java. You can test it directly from the command line using the batch file BuildAndRun.bat, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

© 2015 Altova GmbH

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`AltovaAutomation.dll`	Java-COM bridge: DLL part (for the 32-bit installation)
`AltovaAutomation_x64.dll`	Java-COM bridge: DLL part (for the 64-bit installation)
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`AuthenticActiveX.jar`	Java classes of the Authentic Desktop ActiveX control
`AuthenticDesktopContainer.java`	Java example source code
`AuthenticDesktopContainerEventHandler.java`	Java example source code
`XMLTreeDialog.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`XMLSpyActiveX_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

What the example does

The example places one XMLSpy document editor window, the XMLSpy project window, the XMLSpy XPath window and an XMLSpy entry helper in an AWT frame window. It reads out the File menu defined for XMLSpy and creates an AWT menu with the same structure. You can use this menu or the project window to open and work with files in the document editor.

You can modify the example in any way you like.

The following specific features are described in code listings:

- Creating the ActiveX Controls: Starts Authentic Desktop, which is registered as an automation server, or activates Authentic Desktop if it is already running.
- Loading Data in the Controls: Locates one of the example documents installed with Authentic Desktop and opens it.
- Basic Event Handling: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- UI Update Event Handling: Shows how to handle Authentic Desktop events.
- Creating an XML Tree: Shows how to create an XML tree and prepare it for modal

activation.

Running the example from the command line

Open a command prompt window and type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a Java Development Kit (JDK) 7 or later installation on your computer.

Press **Return**. The Java source in `AuthenticDesktopContainer.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **Import | Existing Projects into Workspace** command to add the Eclipse project file located in the same folder as this Readme file to your workspace. Since you may not have write-access in this folder it is recommended to tell Eclipse to copy the project files into its workspace. The project `AuthenticDesktopContainer` will then appear in your Package Explorer or Navigator.

If you want to use the 64-bit version of the Authentic Desktop ActiveX control you need to use a 64-bit version of Eclipse.

Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `AuthenticDesktopContainer.java` is listed below with comments.

```
001 // access general JAVA-COM bridge classes
002 import com.altova.automation.libs.*;
003
004 //access AuthenticDesktop Java-COM bridge
005 import com.altova.automation.AuthenticDesktop.*;
006 import
com.altova.automation.AuthenticDesktop.Enums.XMLSpyControlPlaceholderWindow;
007 import
com.altova.automation.AuthenticDesktop.Enums.ICActiveXIntegrationLevel;
008
009 // access AWT components
010 import java.awt.*;
011 import java.awt.event.*;
012
013 import javax.swing.*;
014
015
```

 © 2015 Altova GmbH

```
016 /**
017  * A simple example of a container for Authentic Desktop document-level
integration using Java AWT/Swing.
018  * The application's GUI shows a single document editing window and 3
different tool windows:
019  * The elements entry helper, the info window and the attributes entry
helper window.
020  *
021  * The application's menu gets created by reading it out from the Authentic
Desktop control.
022  *
023  * Communication between the project window and the document editing window
gets established
024  * by the event handler for the onOpenedOrFocused event. See
XMLSpyControlEventsDefaultHandler
025  * for further events.
026  *
027  * Feel free to modify and extend this sample.
028  *
029  * @author Altova GmbH
030  */
031 public class AuthenticDesktopContainer
032 {
033   /**
034    * Authentic Desktop manager control - always needed
035    */
036   public static AuthenticDesktopControl         authenticDesktopControl;
037
038   /**
039    * Authentic Desktop document editing control
040    */
041   public static AuthenticDesktopControlDocument
authenticDesktopDocument;
042
043   /**
044    * Tool windows - Authentic Desktop place-holder controls
045    */
046   private static AuthenticDesktopControlPlaceHolder
authenticDesktopInfoToolWindow;
047   private static AuthenticDesktopControlPlaceHolder
authenticDesktopEHElementToolWindow;
048   private static AuthenticDesktopControlPlaceHolder
authenticDesktopProjectToolWindow;
049
050   /**
051    * The hosting frame
052    */
053   private static Frame                  frame;
054
055
056   /**
057    * Helper function that initializes the Document control
058    */
059   public static void initAuthenticDesktopDocument()
060   {
061     try
062     {
063       if ( authenticDesktopDocument != null )
```

```
064         frame.remove( authenticDesktopDocument );
065       authenticDesktopDocument = new AuthenticDesktopControlDocument();
066       frame.add( authenticDesktopDocument, BorderLayout.CENTER );
067       frame.validate();
068       // move the focus to the document control - used when querying for the
command status while enabling/disabling menu items
069       authenticDesktopDocument.requestFocusInWindow();
070     }
071     catch ( Exception ex )
072     {
073       ex.printStackTrace();
074     }
075   }
076
077   /**
078    * Creation of the tree dialog
079    */
080   private static void createTreeWindow(XMLData rootElement)
081   {
082     XMLTreeDialog dlg = new XMLTreeDialog( rootElement, frame );
083     dlg.pack();
084     dlg.setBounds( 0, 0, 300, 500 );
085     dlg.setLocationRelativeTo(frame);
086     dlg.setVisible( true );
087   }
088
089
090   /**
091    * The main entry point.
092    * Creates the application's frame and the XMLSpy windows.
093    * AuthenticDesktopControl, although not visible, is important for the
coordination between the different ActiveX controls of Authentic Desktop.
094    */
095   public static void main( String[] args )
096   {
097     // in case of severe errors somewhere in the ActiveX controls an
AutomationException gets thrown
098     try
099     {
100       // Create the main frame of the application
101       frame = new Frame( "Java ActiveX host window" );
102       frame.setLayout( new BorderLayout() );
103
104       // Create the set of buttons and arrange them in a panel
105       Dimension btnDim = new Dimension ( 130, 25 );
106       JPanel westPanel = new JPanel();
107       westPanel.setPreferredSize( new Dimension( 140, 400 ) );
108       westPanel.setMaximumSize( new Dimension( 140, 800 ) );
109       westPanel.add( new Label( "Open documents") ) ;
110       JButton btnOpenPxf = new JButton("Open OrgChart");
westPanel.add( btnOpenPxf );  btnOpenPxf.setPreferredSize( btnDim );
111       JButton btnOpenXml = new JButton("Open ExpReport");
westPanel.add( btnOpenXml );  btnOpenXml.setPreferredSize( btnDim );
112       westPanel.add( new Label( "Create/destroy") );
113       JButton btnProject = new JButton("Project window");
westPanel.add( btnProject );  btnProject.setPreferredSize( btnDim );
114       JButton btnXpath = new JButton("Info window");
westPanel.add( btnXpath );    btnXpath.setPreferredSize( btnDim );
115       JButton btnEhattrib = new JButton("Entry helper");
westPanel.add( btnEhattrib ); btnEhattrib.setPreferredSize( btnDim );
```


© 2015 Altova GmbH

```
116       westPanel.add( new Label( "Create menu") );
117       JButton btnMenu = new JButton("Load file menu");
westPanel.add( btnMenu );   btnMenu.setPreferredSize( btnDim );
118       westPanel.add( new Label( "Access XML data") );
119       JButton btnTree = new JButton("Show XML tree");
westPanel.add( btnTree );   btnTree.setPreferredSize( btnDim );
120
121       // Create the XMLSpy ActiveX control; the parameter determines that we
want to place document controls and place-holder
122       // controls individually. It gives us full control over the menu, as
well.
123       authenticDesktopControl = new AuthenticDesktopControl
( ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue() );
124
125       // Panel that will hold the Project/Xpath/Attributes windows
126       final JPanel southPanel = new JPanel();
127
128       frame.add( southPanel,        BorderLayout.SOUTH );
129       frame.add( authenticDesktopControl, BorderLayout.NORTH );
130       frame.add( westPanel,       BorderLayout.WEST );
131       initAuthenticDesktopDocument();
132
133       // Listen in this class to communication events ( e.g.
onOpenedOrFocused is handled )
134       final AuthenticDesktopContainerEventHandler handlerObject = new
AuthenticDesktopContainerEventHandler();
135       authenticDesktopControl.addListener( handlerObject );
136
137       // Prepare a shutdown mechanism
138       frame.addWindowListener( new WindowAdapter()
139       {
140         public void windowClosing( WindowEvent ev )
141         {
142           frame.dispose();
143           System.exit( 0 );
144         }
145       } );
146       frame.setVisible( true );
147
148       // Locate samples installed with the product.
149       final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\
\My Documents\\Altova\\Authentic2013\\AuthenticExamples\\";
150
151       // Create a project window and open the sample project in it
152       authenticDesktopProjectToolWindow = new
AuthenticDesktopControlPlaceHolder( XMLSpyControlPlaceholderWindow.XMLSpyControl
ProjectWindowToolWnd.getValue() );
153       authenticDesktopProjectToolWindow.setPreferredSize( new
Dimension( 200, 200 ) );
154       // For the beginning hide the project window
155       authenticDesktopProjectToolWindow.setVisible( false ) ;
156       frame.add( authenticDesktopProjectToolWindow, BorderLayout.NORTH ) ;
157       authenticDesktopProjectToolWindow.openProject( strExamplesFolder  +
"Examples.spp" );
158
159       // Open the PXF file when button is pressed
160       btnOpenPxf.addActionListener( new ActionListener() {
161         public void actionPerformed(ActionEvent e) {
162           try {
163             authenticDesktopControl.open( strExamplesFolder +
```

```
"OrgChart.pxf" );
164           } catch (AutomationException e1) {
165             e1.printStackTrace();
166           }
167         }
168       } );
169
170       // Open the XML file when button is pressed
171       btnOpenXml.addActionListener( new ActionListener() {
172         public void actionPerformed(ActionEvent e) {
173           try {
174             authenticDesktopControl.open( strExamplesFolder +
"ExpReport.xml" );
175           } catch (AutomationException e1) {
176             e1.printStackTrace();
177           }
178         }
179       } );
180
181       // Show/hide the project window when button is pressed
182       btnProject.addActionListener( new ActionListener() {
183         public void actionPerformed(ActionEvent e) {
184           if ( !authenticDesktopProjectToolWindow.isVisible() ) {
185             // remove the hidden window from the frame (which acted like a
temporary parent)
186             frame.remove( authenticDesktopProjectToolWindow ) ;
187             southPanel.add( authenticDesktopProjectToolWindow );
188             authenticDesktopProjectToolWindow.setVisible( true ) ;
189           } else {
190             authenticDesktopProjectToolWindow.setVisible( false ) ;
191             southPanel.remove( authenticDesktopProjectToolWindow );
192             // Add the hidden window to the frame (temporary parent)
193             frame.add( authenticDesktopProjectToolWindow, BorderLayout.NORTH
) ;
194           }
195           frame.validate();
196         }
197       } ) ;
198
199       // Create/destroy the Info window when button is pressed
200       btnXpath.addActionListener( new ActionListener() {
201         public void actionPerformed(ActionEvent e) {
202           if ( authenticDesktopInfoToolWindow == null ) {
203             try {
204               // Create a new window and add it to the south panel
205               authenticDesktopInfoToolWindow = new
AuthenticDesktopControlPlaceHolder( XMLSpyControlPlaceholderWindow.XMLSpyControl
InfoToolWnd.getValue() );
206               authenticDesktopInfoToolWindow.setPreferredSize( new
Dimension( 200, 200 ) );
207               southPanel.add( authenticDesktopInfoToolWindow );
208             } catch (AutomationException e1) {
209               e1.printStackTrace();
210             }
211           } else {
212             // Remove the window and the reference
213             southPanel.remove( authenticDesktopInfoToolWindow );
214             authenticDesktopInfoToolWindow = null;
215           }
216           frame.validate();
```


© 2015 Altova GmbH

```
217         }
218       } ) ;
219
220       // Create/destroy the Entry helper window when button is pressed
221       btnEhattrib.addActionListener( new ActionListener() {
222         public void actionPerformed(ActionEvent e) {
223           if ( authenticDesktopEHElementToolWindow == null ) {
224             try {
225               // Create a new window and add it to the south panel
226               authenticDesktopEHElementToolWindow = new
AuthenticDesktopControlPlaceHolder( XMLSpyControlPlaceholderWindow.XMLSpyControl
EntryHelperTopToolWnd.getValue() );
227               authenticDesktopEHElementToolWindow.setPreferredSize( new
Dimension( 200, 200 ) );
228               southPanel.add( authenticDesktopEHElementToolWindow );
229             } catch (AutomationException e1) {
230               e1.printStackTrace();
231             }
232           } else {
233             southPanel.remove( authenticDesktopEHElementToolWindow );
234             authenticDesktopEHElementToolWindow = null;
235           }
236           frame.validate();
237         }
238       } ) ;
239
240       // Load the file menu when the button is pressed
241       btnMenu.addActionListener( new ActionListener() {
242         public void actionPerformed(ActionEvent e) {
243           try {
244             // Create the menubar that will be attached to the frame
245             MenuBar mb = new MenuBar();
246             // Load the main menu's first item - the File menu
247             XMLSpyCommand xmlSpyMenu =
authenticDesktopControl.getMainMenu().getSubCommands().getItem( 0 );
248             // Create Java menu items from the Commands objects
249             Menu fileMenu = new Menu();
250             handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
251             fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
252             mb.add( fileMenu );
253             frame.setMenuBar( mb );
254             frame.validate();
255           } catch (AutomationException e1) {
256             e1.printStackTrace();
257           }
258           // Disable the button when the action has been performed
259           ((AbstractButton) e.getSource()).setEnabled( false );
260         }
261       } ) ;
262
263       // Load a tree when the button is pushed
264       btnTree.addActionListener( new ActionListener() {
265         public void actionPerformed(ActionEvent e) {
266           try {
267
          createTreeWindow( authenticDesktopDocument.getDocument().getRootElem
ent() );
268           } catch (AutomationException e1) {
269             e1.printStackTrace();
270           }
```

```
271         }
272       } ) ;
273
274       // Show the main window
275       frame.setBounds( 0, 0, 800, 600 );
276       frame.validate();
277
278       // feel free to extend.
279     }
280     catch ( AutomationException e )
281     {
282       e.printStackTrace();
283     }
284   }
285
286 }
```

Creating the ActiveX Controls

The code listing below show how ActiveX controls can be created. The constructors will create the Java wrapper objects. Adding these Canvas-derived objects to a panel or to a frame will trigger the creation of the wrapped ActiveX object.

```
01   /**
02    * Authentic Desktop manager control - always needed
03    */
04   public static AuthenticDesktopControl         authenticDesktopControl =
null;
05
06   /**
07    * Authentic Desktop document editing control
08    */
09   public static AuthenticDesktopControlDocument     authenticDesktopDocument
= null;
10
11   /**
12    * Tool windows - Authentic Desktop place-holder controls
13    */
14   private static AuthenticDesktopControlPlaceHolder
authenticDesktopInfoToolWindow = null;
15   private static AuthenticDesktopControlPlaceHolder
authenticDesktopEHElementToolWindow = null;
16   private static AuthenticDesktopControlPlaceHolder
authenticDesktopProjectToolWindow = null;
17
18   // Create the Authentic Desktop ActiveX control, The parameter determines
that we want
     // to place document controls and place-holder controls individually.
19   // It gives us full control over the menu, as well.
20      authenticDesktopControl = new AuthenticDesktopControl(
        ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue()
);
21
22      authenticDesktopDocument = new AuthenticDesktopControlDocument();
23      authenticDesktopDocument.setPreferredSize( new Dimension ( 640, 480 ) );
```


© 2015 Altova GmbH

```
24      frame.add( authenticDesktopDocument, BorderLayout.CENTER );
25
26   // Create a project window and open the sample project in it
27      authenticDesktopProjectToolWindow = new
AuthenticDesktopControlPlaceHolder(
        XMLSpyControlPlaceholderWindow.XMLSpyControlProjectWindowToolWnd.getValu
e() );
28      authenticDesktopProjectToolWindow.setPreferredSize( new Dimension( 200,
200 ) );
```

Loading Data in the Controls

The code listing below show how data can be loaded in the ActiveX controls.

```
1    // Locate samples installed with the product.
2       final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
        "\\My Documents\\Altova\\Authentic2015\\AuthenticExamples\\";
3       authenticDesktopProjectToolWindow = new
AuthenticDesktopControlPlaceHolder( XMLSpyControlPlaceholderWindow.XMLSpyControl
ProjectWindowToolWnd.getValue() );
```

Basic Event Handling

The code listing below shows how basic events can be handled. When calling the AuthenticDesktopControl's `open` method, or when trying to open a file via the menu or Project tree, the `onOpenedOrFocused` event is sent to the attached event handler. The basic handling for this event is opening the file by calling the Authentic DesktopDocumentControl's `open` method.

```
01 // Open the PXF file when button is pressed
02       btnOpenPxf.addActionListener( new ActionListener() {
03         public void actionPerformed(ActionEvent e) {
04           try {
05             authenticDesktopControl.open( strExamplesFolder +
"OrgChart.pxf" );
06           } catch (AutomationException e1) {
07             e1.printStackTrace();
08           }
09         }
10       } );
11       public void onOpenedOrFocused( String i_strFileName, boolean
i_bOpenWithThisControl, boolean i_bFileAlreadyOpened ) throws
AutomationException
12   {
13     // Handle the New/Open events coming from the Project tree or from the
menus
14     if ( !i_bFileAlreadyOpened )
15     {
16       // This is basically an SDI interface, so open the file in the already
existing document control
17       try {
18
        AuthenticDesktopContainer.authenticDesktopDocument.open( i_strFileName )
;
19
```

```
        AuthenticDesktopContainer.authenticDesktopDocument.requestFocusInWindow(
);
        } catch (Exception e) {
          e.printStackTrace();
        }
      }
    }
```

Menus

The code listing below shows how menu items can be created. Each `AuthenticDesktopCommand` object gets a corresponding `MenuItem` object, with the `ActionCommand` set to the ID of the command. The actions generated by all menu items are handled by the same function, which can perform specific handlings (like reinterpreting the closing mechanism) or can delegate the execution to the `AuthenticDesktopControl` object by calling its `exec` method. The `menuMap` object that is filled during menu creation is used later (see section UI Update Event Handling).

```
01  // Load the file menu when the button is pressed
02       btnMenu.addActionListener( new ActionListener() {
03         public void actionPerformed(ActionEvent e) {
04           try {
05             // Create the menubar that will be attached to the frame
06             MenuBar mb = new MenuBar();
07             // Load the main menu's first item - the File menu
08             XMLSpyCommand xmlSpyMenu =
xmlSpyControl.getMainMenu().getSubCommands().getItem( 0 );
09             // Create Java menu items from the Commands objects
10             Menu fileMenu = new Menu();
11             handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
12             fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
13             mb.add( fileMenu );
14             frame.setMenuBar( mb );
15             frame.validate();
16           } catch (AutomationException e1) {
17             e1.printStackTrace();
18           }
19           // Disable the button when the action has been performed
20           ((AbstractButton) e.getSource()).setEnabled( false );
21         }
22       } ) ;
23 /** * Populates a menu with the commands and submenus contained in an
XMLSpyCommands object  */
24       public void fillMenu(Menu newMenu, XMLSpyCommands xmlSpyMenu) throws
AutomationException
25   {
26     // For each command/submenu in the xmlSpyMenu
27     for ( int i = 0 ; i < xmlSpyMenu.getCount() ; ++i  )
28     {
29       XMLSpyCommand xmlSpyCommand = xmlSpyMenu.getItem( i );
30       if ( xmlSpyCommand.getIsSeparator() )
31         newMenu.addSeparator();
32       else
33       {
34         XMLSpyCommands subCommands = xmlSpyCommand.getSubCommands();
35         // Is it a command (leaf), or a submenu?
```

 © 2015 Altova GmbH

```
36          if ( subCommands.isNull() || subCommands.getCount() == 0 )
37          {
38            // Command -> add it to the menu, set its ActionCommand to its ID
and store in in the menuMap
39            MenuItem mi = new MenuItem( xmlSpyCommand.getLabel().replace( "&",
"" ) );
40            mi.setActionCommand( "" + xmlSpyCommand.getID() );
41            mi.addActionListener( this );
42            newMenu.add( mi );
43            menuMap.put( xmlSpyCommand.getID(), mi );
44          }
45          else
46          {
47            // Submenu -> create submenu and repeat recursively
48            Menu newSubMenu = new Menu();
49            fillMenu( newSubMenu, subCommands );
50            newSubMenu.setLabel( xmlSpyCommand.getLabel().replace( "&", "" ) );
51            newMenu.add( newSubMenu );
52          }
53        }
54      }
55    }
56
57     /**
58     * Action handler for the menu items
59     * Called when the user selects a menu item; the item's action command
corresponds to the command table for XMLSpy
60     */
61    public void actionPerformed( ActionEvent e )
62    {
63      try
64      {
65        int iCmd = Integer.parseInt( e.getActionCommand() );
66        // Handle explicitly the Close commands
67        switch ( iCmd )
68        {
69          case 57602:       // Close
70          case 34050:       // Close All
71            AuthenticDesktopContainer.initXmlSpyDocument();
72            break;
73          default:
74            AuthenticDesktopContainer.xmlSpyControl.exec( iCmd );
75            break;
76        }
77      }
78      catch ( Exception ex )
79      {
80        ex.printStackTrace();
81      }
82
83    }
```

UI Update Event Handling

The code listing below shows how a UI-Update event handler can be created.

```
01     /**
02     * Call-back from the XMLSpyControl.
```

```
03     * Called to enable/disable commands
04     */
05    @Override
06    public void onUpdateCmdUI() throws AutomationException
07    {
08      // A command should be enabled if the result of queryStatus contains the
Supported (1) and Enabled (2) flags
09      for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10
       pair.getValue().setEnabled( AuthenticDesktopContainer.authenticDesktopCont
rol.queryStatus( pair.getKey() ) > 2 );
11    }
12    /**
13     * Call-back from the XMLSpyControl.
14     * Usually called while enabling/disabling commands due to UI updates
15     */
16    @Override
17    public boolean onIsActiveEditor( String i_strFilePath ) throws
AutomationException
18    {
19      try {
20        return
AuthenticDesktopContainer.authenticDesktopDocument.getDocument().getFullName().e
qualsIgnoreCase( i_strFilePath );
21      } catch ( Exception e ) {
22        return false;
23      }
24    }
```

Creating an XML Tree

The listing below loads an XML data object as nodes in a tree.

```
01 // access required XMLSpy Java-COM classes
02 import com.altova.automation.XMLSpy.XMLData;
03
04 // access AWT and Swing components
05 import java.awt.*;
06 import javax.swing.*;
07 import javax.swing.tree.*;
08
09 /**
10  * A simple example of a tree control loading the structure from an XMLData
object.
11  * The class receives an XMLData object, loads its nodes in a JTree, and
prepares
12  * for modal activation.
13  *
14  * Feel free to modify and extend this sample.
15  *
16  * @author Altova GmbH
17  */
18 class XMLTreeDialog extends JDialog
19 {
20   /**
21    * The tree control
```


© 2015 Altova GmbH

```
22     */
23    private JTree myTree;
24
25    /**
26     * Root node of the tree control
27     */
28    private DefaultMutableTreeNode top ;
29
30    /**
31     * Constructor that prepares the modal dialog containing the filled tree
control
32     * @param xml    The data to be displayed in the tree
33     * @param parent  Parent frame
34     */
35    public XMLTreeDialog( XMLData xml, Frame parent )
36    {
37      // Construct the modal dialog
38      super( parent, "XML tree", true );
39      // Arrange controls in the dialog
40      top = new DefaultMutableTreeNode("root");
41      myTree = new JTree(top);
42      setContentPane( new JScrollPane( myTree ) );
43      // Build up the tree
44      fillTree( top, xml );
45      myTree.expandRow( 0 );
46    }
47
48    /**
49     * Loads the nodes of an XML element under a given tree node
50     * @param node  Target tree node
51     * @param elem  Source XML element
52     */
53    private void fillTree( DefaultMutableTreeNode node, XMLData elem)
54    {
55      try
56      {
57        // There are several ways to iterate through child elements: either
using the getFirstChild/getNextChild,
58        // or by incrementing an index up to countChildren and calling getChild
[as shown below].
59        // If you only want to get childer of one kind, you should use
countChildrenKind/getChildKind,
60        // or provide a kind to the getFirstChild before iterating with the
getNextChild.
61        int nSize = elem.countChildren() ;
62        for ( int i = 0 ; i < nSize ; ++i)
63        {
64          // Create a new tree node for each child element, and continue
recursively
65          XMLData newElem = elem.getChild(i) ;
66          DefaultMutableTreeNode newNode = new
DefaultMutableTreeNode( newElem.getName() ) ;
67          node.add( newNode ) ;
68          fillTree( newNode, newElem ) ;
69        }
70      }
71      catch (Exception e)
72      {
73        e.printStackTrace();
```

```
74          }
75      }
76
77  }
```


© 2015 Altova GmbH

4.4 Command Table for Authentic Desktop

Tables in this section list the names and identifiers of all commands that are available within Authentic Desktop. Every sub-section lists the commands from the corresponding top-level menu of Authentic Desktop. The left-most column shows the command's menu text to make it easier for you to identify the functionality behind the command. The last sub-section is a collection of those commands that are not accessible via the main menu.

See Query Authentic Desktop Commands on how to query the current resource structure and command availability.

Use the command identifiers with AuthenticDesktopControl.QueryStatus or AuthenticDesktopControlDocument.QueryStatus to check the current status of a command. Use AuthenticDesktopControl.Exec or AuthenticDesktopControlDocument.Exec to execute a command.

File Menu
Edit Menu
Project Menu
XML Menu
XSL/XQuery Menu
Authentic Menu
View Menu
Browser Menu
Tools Menu
Window Menu
Help Menu
Misc Menu

4.4.1 File Menu

File menu commands:

New...	ID_FILE_NEW	57600
Open...	ID_FILE_OPEN	57601
Open Global Resource...	IDC_OPEN_GLOBALRESOURCE	34112
Reload	IDC_FILE_RELOAD	34065
Encoding...	IDC_ENCODING	34061
Close	ID_FILE_CLOSE	57602
Close All	IDC_CLOSE_ALL	34050
Save	ID_FILE_SAVE	57603
Save As...	ID_FILE_SAVE_AS	57604
Save to URL...	ID_FILE_SAVE_TO_URL	34209
Save All	ID_FILE_SAVE_ALL	34208
Send by Mail...	ID_FILE_SEND_MAIL	57612
Print...	ID_FILE_PRINT	57607
Print Preview	IDC_PRINT_PREVIEW	34104
Print Setup...	ID_FILE_PRINT_SETUP	57606
Recent File	ID_FILE_MRU_FILE1	57616
Exit	ID_APP_EXIT	57665

© 2015 Altova GmbH

4.4.2 Edit Menu

Edit menu commands:

Undo	ID_EDIT_UNDO	57643
Redo	ID_EDIT_REDO	57644
Cut	ID_EDIT_CUT	57635
Copy	ID_EDIT_COPY	57634
Paste	ID_EDIT_PASTE	57637
Delete	ID_EDIT_CLEAR	57632
Select All	ID_EDIT_SELECT_ALL	57642
Find...	ID_EDIT_FIND	57636
Find next	ID_EDIT_REPEAT	57640
Replace...	ID_EDIT_REPLACE	57641

4.4.3 Project Menu

Project menu commands:

New Project	IDC_ICPROJECTGUI_NEW	37200
Open Project...	IDC_ICPROJECTGUI_OPEN	37201
Reload Project	IDC_ICPROJECTGUI_RELOAD	37202
Close Project	IDC_ICPROJECTGUI_CLOSE	37203
Save Project	IDC_ICPROJECTGUI_SAVE	37204
Source Control/Open Project...	IDC_SCC_OPEN_PROJECT	34140
Source Control/Enable Source Code Control	IDC_SCC_ENABLE	34137
Source Control/Get latest version...	IDC_SCC_GET	34138
Source Control/Check Out...	IDC_SCC_CHECK_OUT	34135
Source Control/Check In...	IDC_SCC_CHECK_IN	34134
Source Control/Undo Check Out...	IDC_SCC_UNDO_CHECK_OUT	34145
Source Control/Add to Source Control	IDC_SCC_ADD	34133
Source Control/Remove from Source Control	IDC_SCC_REMOVE	34143
Source Control/Show History...	IDC_SCC_HISTORY	34139
Source Control/Show Differences...	IDC_SCC_DIFF	34136
Source Control/Properties...	IDC_SCC_PROPERTIES	34141
Source Control/Refresh Status...	IDC_SCC_REFRESH	34142
Source Control/Run Native Interface...	IDC_SCC_RUN	34144
Add Files to Project...	IDC_ICPROJECTGUI_ADD_FILES_TO_PROJECT	37205
Add Global Resource to Project...	IDC_ICPROJECTGUI_ADD_GLOBAL_RESOURCE_TO_PROJECT	37239
Add URL to Project...	IDC_ICPROJECTGUI_ADD_URL_TO_PROJECT	37206
Add Active File to Project	IDC_ICPROJECTGUI_ADD_ACTIVE_FILE_TO_PROJECT	37208
Add Active and Related Files to Project	IDC_ICPROJECTGUI_ADD_ACTIVE_AND_RELATED_FILES_TO_PROJECT	37209
Add Project Folder to Project...	IDC_ICPROJECTGUI_ADD_FOLDER_TO_PROJECT	37210
Add External Folder to Project...	IDC_ICPROJECTGUI_ADD_EXT_FOLDER_TO_PROJECT	37211
Add External Web Folder to Project...	IDC_ICPROJECTGUI_ADD_EXT_URL_FOLDER_TO_PROJECT	37212
Properties...	IDC_ICPROJECTGUI_PROJECT_PROPERTIES	37223
Recent Project	IDC_ICPROJECTGUI_RECENT	37224

© 2015 Altova GmbH

4.4.4 XML Menu

XML menu commands:

Check well-formedness	IDC_CHECK_WELL_FORM	34049
Validate XML	IDC_VALIDATE	32954

© 2015 Altova GmbH

4.4.5 XSL/XQuery Menu

XSL/XQuery menu commands:

XSL Transformation	IDC_TRANSFORM_XSL	33006
XSL:FO Transformation	IDC_TRANSFORM_XSLFO	33007
XSL Parameters/XQuery Variables...	IDC_TRANSFORM_XSL_PARAMS	33008

4.4.6 Authentic Menu

Authentic menu commands:

New Document...	IDC_AUTHENTIC_NEW_FILE	34036
Edit Database Data...	IDC_AUTHENTIC_EDIT_DB	34035
Edit StyleVision Stylesheet	IDC_EDIT_SPS	34060
Select new row with XML data for editing	IDC_CHANGE_WORKING_DB_XML_CELL	32861
Define XML Entities...	IDC_DEFINE_ENTITIES	32805
Hide markup	IDC_MARKUP_HIDE	32855
Show Small markup	IDC_MARKUP_SMALL	32858
Show Large markup	IDC_MARKUP_LARGE	32856
Show Mixed markup	IDC_MARKUP_MIX	32857
Append row	IDC_ROW_APPEND	32806
Insert row	IDC_ROW_INSERT	32809
Duplicate row	IDC_ROW_DUPLICATE	32808
Move row Up	IDC_ROW_MOVE_UP	32811
Move row Down	IDC_ROW_MOVE_DOWN	32810
Delete row	IDC_ROW_DELETE	32807

© 2015 Altova GmbH

4.4.7 View Menu

View menu commands:

WSDL Design view	IDC_VIEW_WSDL	34117
XBRL Taxonomy view	IDC_VIEW_XBRL	34118
Authentic view	IDC_VIEW_CONTENT	34177
Browser view	IDC_VIEW_BROWSER	34176
Text View Settings	IDC_TEXTVIEW_SETTINGS	34119

4.4.8 Browser Menu

Browser menu commands:

Back	IDC_BROWSER_BACK	34039
Forward	IDC_BROWSER_FORWARD	34045
Stop	IDC_BROWSER_STOP	34047
Refresh	IDC_BROWSER_REFRESH	34046
Fonts/Largest	IDC_BROWSER_FONT_LARGEST	34041
Fonts/Larger	IDC_BROWSER_FONT_LARGE	34040
Fonts/Medium	IDC_BROWSER_FONT_MEDIUM	34042
Fonts/Smaller	IDC_BROWSER_FONT_SMALL	34043
Fonts/Smallest	IDC_BROWSER_FONT_SMALLEST	34044
Separate window	IDC_BROWSER_USE_OWN_FRAME	34048

© 2015 Altova GmbH

4.4.9 Tools Menu

Tools menu command:

Spelling...	IDC_SPELL_CHECK	34154
Spelling options...	IDC_SPELL_OPTIONS	34155
Switch to Scripting environment...	ID_WINDOW_SWITCHTOVBA	34231
Show macros...	ID_WINDOW_VBAMACROS	34232
Project/Assign Scripts to Project	ID_SCRIPTINGPRJ_ASSIGN	34214
Project/Unassign Scripts from Project	ID_SCRIPTINGPRJ_UNASSIGN	34215
Project/Project Scripts active	ID_SCRIPTINGPRJ_ACTIVE	34213
Global Resources	IDC_GLOBALRESOURCES	37401
Active Configuration/<plugin not loaded>	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Customize...	IDC_CUSTOMIZE	34055
Options...	IDC_SETTINGS	33300
<placeholder>	ID_SCRIPTING_MACROITEMS	34249

4.4.10 Window Menu

Window menu commands:

Cascade	ID_WINDOW_CASCADE	57650
Tile horizontally	ID_WINDOW_TILE_HORZ	57651
Tile vertically	ID_WINDOW_TILE_VERT	57652
Project window	IDC_PROJECT_WINDOW	34128
Info window	IDC_INFO_WINDOW	34085
Entry Helpers	IDC_ENTRY_HELPERS	34062
Output windows	IDC_OUTPUT_DIALOGBARS	34004
Project and Entry Helpers	IDC_PROJECT_ENTRYHELPERS	34006
All on/off	IDC_ALL_BARS	34031

© 2015 Altova GmbH

4.4.11 Help Menu

Help menu commands:

Table of Contents...	IDC_HELP_CONTENTS	34076
Index...	IDC_HELP_INDEX	34077
Search...	IDC_HELP_SEARCH	34079
Keyboard Map...	IDC_HELP_KEYMAPDLG	34078
Software Activation...	IDC_ACTIVATION	34005
Order Form...	IDC_OPEN_ORDER_PAGE	34094
Registration...	IDC_REGISTRATION	34131
Check for Updates...	IDC_CHECK_FOR_UPDATES	34275
Support Center...	IDC_OPEN_SUPPORT_PAGE	34096
FAQ on the Web...	IDC_SHOW_FAQ	34153
Download Components and Free Tools	IDC_OPEN_COMPONENTS_PAGE	34093
Authentic Desktop on the Internet..	IDC_OPEN_AUTHENTIC_HOME	34098
Authentic Desktop Training...	ID_HELP_AUTHENTICTRAINING	34210
About Authentic Desktop	ID_APP_ABOUT	57664

4.4.12 Misc Menu

Miscellaneous menu and context menu commands:

Database	IDC_ADDRESOURCE_DATABASE	37405
File	IDC_ADDRESOURCE_FILE	37403
Folder	IDC_ADDRESOURCE_FOLDER	37404
Attribute	IDC_ADD_CHILD_ATTRIBUTE	33402
CData	IDC_ADD_CHILD_CDATA	33403
Comment	IDC_ADD_CHILD_COMMENT	33404
ATTLIST	IDC_ADD_CHILD_DEF_ATTLIST	33405
DOCTYPE	IDC_ADD_CHILD_DEF_DOCTYPE	33406
ELEMENT	IDC_ADD_CHILD_DEF_ELEMENT	33407
ENTITY	IDC_ADD_CHILD_DEF_ENTITY	33408
ExternalID	IDC_ADD_CHILD_DEF_EXTERNAL_ID	33409
NOTATION	IDC_ADD_CHILD_DEF_NOTATION	33410
Processing Instruction	IDC_ADD_CHILD_PI	33411
Element	IDC_ADD_CHILD_STRUCT	33412
Text	IDC_ADD_CHILD_TEXT	33413
XInclude...	IDC_ADD_CHILD_XINCLUDE	34027
XML	IDC_ADD_CHILD_XML	33414
Attribute	IDC_APPEND_ATTRIBUTE	33415
CData	IDC_APPEND_CDATA	33416
Comment	IDC_APPEND_COMMENT	33417
ATTLIST	IDC_APPEND_DEF_ATTLIST	33418
DOCTYPE	IDC_APPEND_DEF_DOCTYPE	33419
ELEMENT	IDC_APPEND_DEF_ELEMENT	33420
ENTITY	IDC_APPEND_DEF_ENTITY	33421
ExternalID	IDC_APPEND_DEF_EXTERNAL_ID	33422

contd...

NOTATION	IDC_APPEND_DEF_NOTATION	33423
Processing Instruction	IDC_APPEND_PI	33424
Element	IDC_APPEND_STRUCT	33425
Text	IDC_APPEND_TEXT	33426

© 2015 Altova GmbH

XInclude...	IDC_APPEND_XINCLUDE	34026
XML	IDC_APPEND_XML	33427
Assign DTD...	IDC_ASSIGN_DTD	34032
Assign sample XML file...	IDC_ASSIGN_SAMPLE_XML	33000
Assign Schema...	IDC_ASSIGN_SCHEMA	34033
Assign a StyleVision Stylesheet...	IDC_ASSIGN_SPS	34034
Assign XSL...	IDC_ASSIGN_XSL	33001
Assign XSL:FO...	IDC_ASSIGN_XSLFO	33002
Collapse unselected	IDC_COLLAPSE_UNSELECTED	33428
Convert all to Global Resources	IDC_CONVERT_ALL_DATASOURCES_TO_GLOBAL_RESOURCES	36687
Convert to Global Resource	IDC_CONVERT_DATASOURCE_TO_GLOBAL_RESOURCES	36684
Map to other DTD/Schema or DB in MapForce	IDC_DTD_OPENIN_MAPFORCE	34056
Design HTML/PDF Output in StyleVision	IDC_DTD_OPENIN_STYLEVISION	34057
Edit Global Resource...	IDC_EDIT_GLOBAL_RESOURCES	36686
Insert file path...	IDC_EDIT_INSERT_PATH_STRING	34013
Insert XInclude...	IDC_EDIT_INSERT_XINCLUDE_STRING	34017
Enclose in Element	IDC_ENCLOSE_IN_ELEMENT	33446
Check In...	IDC_FILE_CHECK_IN	32951
Check Out...	IDC_FILE_CHECK_OUT	32952
Undo Check Out...	IDC_FILE_UNDO_CHECK_OUT	32953
Find in files...	IDC_FIND_IN_FILES	34000

contd...

Flush memory cache	IDC_FLUSH_CACHED_FILES	34066
Generate Program Code...	IDC_GENERATE_CODE_FROM_SCHEMA	34067
Browse...	IDC_GLOBALRESOURCESUI_CHOOSEFILEASFILE	37420
Choose another Global Resource...	IDC_GLOBALRESOURCESUI_CHOOSEFILEASGR	37419
Add configuration	IDC_GLOBALRESOURCESUI_DETAILS_ADDCONFIG	37423
Add a configuration copy	IDC_GLOBALRESOURCESUI_DETAILS_ADDCONFIGCOPY	37424
Delete configuration	IDC_GLOBALRESOURCESUI_DETAILS_DELETECONFIG	37425
Active Global Resource Configuration	IDC_GLOBALRESOURCES_ACTIVECONFIG	37400
Delete	IDC_GLOBALRESOURCES_MAINDLG_DELETERESOURCE	37422

Go to Next Bookmark	IDC_GOTONEXTBOOKMARK	34070
Go to Previous Bookmark	IDC_GOTOPREVBOOKMARK	34071
Go to Definition	IDC_GOTO_DEFINITION	33447
Go to DTD	IDC_GOTO_DTD	34072
Go to File	IDC_GOTO_FILE	33448
Go to line/char	IDC_GOTO_LINE	34073
Go to Schema	IDC_GOTO_SCHEMA	34074
Go to XSL	IDC_GOTO_XSL	33004
Display as Table	IDC_GRID_VIEW_AS_TABLE	34075
Edit a SELECT Statement...	IDC_ICDBWNDS_EDIT_LOCALVIEW	36676
Remove SELECT statement	IDC_ICDBWNDS_REMOVE_LOCALVIEW	36680
Add a SELECT Statement...	IDC_ICDBWND_CREATE_LOCALVIEW	36677
Generate and add a SELECT Statement...	IDC_ICDBWND_GENEATE_SQL_FOR_LOCALVIEW	36678
Remove from Online Browser	IDC_ICDBWND_REMOVE_FROM_BROWSERVIEW	36683
Add Active and Related Files	IDC_ICPROJECTGUI_ADD_ACTIVE_AND_RELATED_FILES	37217
Add Active File	IDC_ICPROJECTGUI_ADD_ACTIVE_FILE	37216

contd...

Add External Folder...	IDC_ICPROJECTGUI_ADD_EXT_FOLDER	37219
Add External Web Folder...	IDC_ICPROJECTGUI_ADD_EXT_URL_FOLDER	37220
Add Files...	IDC_ICPROJECTGUI_ADD_FILES	37213
Add Project Folder...	IDC_ICPROJECTGUI_ADD_FOLDER	37218
Add Global Resource...	IDC_ICPROJECTGUI_ADD_GLOBAL_RESOURCE	37238
Add URL...	IDC_ICPROJECTGUI_ADD_URL	37214
Properties...	IDC_ICPROJECTGUI_PROPERTIES	37222
Refresh	IDC_ICPROJECTGUI_REFRESH_EXT_FOLDER	37221
Include another DTD...	IDC_INCLUDE_DTD	34084
Move left	IDC_MOVE_LEFT	34091
Move right	IDC_MOVE_RIGHT	34092
Namespace prefix...	IDC_NAMESPACE	33462
Optimal widths	IDC_OPTIMAL_WIDTHS	34099
0 or More Matches	IDC_RECM_0MATCH	33901

© 2015 Altova GmbH

1 or More Matches	IDC_RECM_1MATCH	33902
Any Character	IDC_RECM_ANYCHAR	33903
Character in Range	IDC_RECM_CHARINRANGE	33904
Character Not in Range	IDC_RECM_CHARNOTINRANGE	33905
End of Line	IDC_RECM_ENDLINE	33906
End of Word	IDC_RECM_ENDWORD	33907
Beginning of Line	IDC_RECM_STARTLINE	33908
Beginning of Word	IDC_RECM_STARTWORD	33909
Tagged Expression	IDC_RECM_TAGGEDEXP	33910
Collapse -	IDC_SEL_COLLAPSE	34151
Expand +	IDC_SEL_EXPAND	34152
Expand fully	IDC_SEL_EXPAND_ALL	33463

contd...

Append Row	IDC_TABLE_APPEND_ROW	34157
Insert Row	IDC_TABLE_INSERT_ROW	34158
Ascending Sort	IDC_TABLE_SORT_ASC	33464
Descending Sort	IDC_TABLE_SORT_DESC	33465
Insert/Remove Bookmark	IDC_TOGGLE_BOOKMARK	34162
Comment In/Out	IDC_TOGGLE_XML_COMMENT	34029
Update Entry Helpers	IDC_UPDATE_ELEMENT_CHOICE	34173
Messages window	IDC_VALIDATOR_OUTPUT	34175
Word Wrap	IDC_WORD_WRAP	34181
Add Active File to Project	ID_ADDACTIVEFILETOPROJECT	36549
Add Files to Project ...	ID_ADDFILESTOPROJECT	36550
Create Folder	ID_CREATEFOLDER	36555
Disconnect	ID_DISCONNECT	36559
Disconnect from all Data Sources	ID_DISCONNECTFROMALLDATASOURCES	36560
Edit Data	ID_EDITRESULTDATA	36646
Rename	ID_EDIT_RENAME	36563
Open Project	ID_FILE_LOAD_PROJECT	36565
Adds a configuration	ID_GLOBALRESOURCES_ADDCONFIG	37429
Adds a configuration as copy of the currently selected configuration	ID_GLOBALRESOURCES_ADDCONFIGCOPY	37430
Deletes a configuration	ID_GLOBALRESOURCES_DELCONFIG	37431
Show referenced table	ID_ICDBWNDS_FGNKEY_GOTO_REFERENCE	36567

View in XMLSpy	ID_ICDBWNDS_VIEW_XMLSCHEMA_IN_XMLSPY	36500
Add a New Data Source	ID_ICDBWND_ADDANEWDATASOURCE	36568
Add to Design Editor	ID_ICDBWND_BROWSER_ADD_TO_DESIGNVIEW	36569
Add to/Remove from Favorites	ID_ICDBWND_BROWSER_ADD_TO_FAVOURITES	36570
Refresh	ID_ICDBWND_BROWSER_REFRESH_ROOT	36571

contd...

All	ID_ICDBWND_BROWSER_SEARCH_MODE_ALL	36575
From current DataSource	ID_ICDBWND_BROWSER_SEARCH_MODE_DATASOURCE	36576
From focused item	ID_ICDBWND_BROWSER_SEARCH_MODE_FOCUSED_ITEM	36577
Show in new Design Editor	ID_ICDBWND_BROWSER_SHOW_IN_DESIGNVIEW	36578
Check	ID_ICDBWND_CHECK	36579
Check Children	ID_ICDBWND_CHECK_ALL	36580
Clear	ID_ICDBWND_CLEAR_ROWCOUNT	36538
Children	ID_ICDBWND_COLLAPSE_CHILDREN	36581
Siblings	ID_ICDBWND_COLLAPSE_SIBLING	36582
Execute SQL	ID_ICDBWND_EXECUTE	36583
Children	ID_ICDBWND_EXPAND_CHILDREN	36584
Siblings	ID_ICDBWND_EXPAND_SIBLINGS	36585
Export database data	ID_ICDBWND_EXPORT	36586
Add	ID_ICDBWND_FILEDSN_ADD	36588
Delete	ID_ICDBWND_FILEDSN_DELETE	36589
Contains	ID_ICDBWND_FILTER_CONTAINS	36591
Does not contain	ID_ICDBWND_FILTER_DOES_NOT_CONTAIN	36592
Ends with	ID_ICDBWND_FILTER_ENDS_WITH	36593
Equals	ID_ICDBWND_FILTER_EQUALS	36594
No Filter	ID_ICDBWND_FILTER_INCACTIVE	36596
Starts with	ID_ICDBWND_FILTER_STARTS_WITH	36597
Connect	ID_ICDBWND_MENU_DATASOURCE_CONNECT	36605
Disconnect	ID_ICDBWND_MENU_DATASOURCE_DISCONNECT	36606
Get Tables	ID_ICDBWND_MENU_DATASOURCE_GETTABLES	36607
Preview	ID_ICDBWND_PREVIEWITEM	36608
Remove all favorites	ID_ICDBWND_REMOVE_ALL_FAVORITES	36609
Toggle	ID_ICDBWND_TOGGLE	36610

© 2015 Altova GmbH

Uncheck	ID_ICDBWND_UNCHECK	36611

contd...

Uncheck Children	ID_ICDBWND_UNCHECK_ALL	36612
Show/Update	ID_ICDBWND_UPDATE_ROWCOUNT	36537
Open	ID_OPEN	36619
Add file extension	ID_POPUP_ADDEXTENSION	35003
Add search path	ID_POPUP_ADDSEARCHPATH	35004
Delete file extension	ID_POPUP_DELETEEXTENSION	35005
Delete search path	ID_POPUP_DELETESEARCHPATH	35006
Edit file extension	ID_POPUP_EDITEXTENSION	35007
Edit search path	ID_POPUP_EDITSEARCHPATH	35008
Reload search paths	ID_POPUP_RELOAD_SEARCH_PATHS	35009
Reset search paths	ID_POPUP_RESET_SEARCH_PATHS	33810
Start Debugger/Go	ID_PROCESS_XSL	34212
New Project	ID_PROJECT_NEWPROJECT	36620
Append	ID_RELDISP_APPEND_LAYER	33201
Delete	ID_RELDISP_DELETE_LAYER	33202
Focus next item on active layer	ID_RELDISP_FOCUS_NEXT_ITEM_ON_LAYER	33227
Focus previous item on active layer	ID_RELDISP_FOCUS_PREV_ITEM_ON_LAYER	33228
Selected Layer	ID_RELDISP_HIDE_LAYER	33203
Others	ID_RELDISP_HIDE_LAYER_OTHER	33204
Insert	ID_RELDISP_INSERT_LAYER	33205
Selected Layer	ID_RELDISP_LOCK_LAYER	33206
Others	ID_RELDISP_LOCK_LAYER_OTHER	33207
Move Back	ID_RELDISP_MOVE_LAYER_ITEMS_BACK	33208
Move Front	ID_RELDISP_MOVE_LAYER_ITEMS_FRONT	33209
Selected Layer	ID_RELDISP_PALE_LAYER	33210

contd...

Others	ID_RELDISP_PALE_LAYER_OTHER	33211
Selected Layer	ID_RELDISP_RECOLOR_LAYER	33212

Others	ID_RELDISP_RECOLOR_LAYER_OTHER	33213
Rename	ID_RELDISP_RENAME_LAYER	33214
Reset all layer states	ID_RELDISP_RESET_LAYER_STATES	33231
Select	ID_RELDISP_SELECT_ITEMS	33215
Select Others	ID_RELDISP_SELECT_OTHER_ITEMS	33216
Selected Layer	ID_RELDISP_SHOW_LAYER	33217
Show layer item count	ID_RELDISP_SHOW_LAYER_ITEM_COUNT	33229
Others	ID_RELDISP_SHOW_LAYER_OTHER	33218
Selected Layer	ID_RELDISP_TOGGLE_COLORING	33219
Others	ID_RELDISP_TOGGLE_COLORING_OTHER	33220
Selected Layer	ID_RELDISP_TOGGLE_LAYER_VISIBILITY	33221
Others	ID_RELDISP_TOGGLE_LAYER_VISIBILITY_OTHER	33222
Selected Layer	ID_RELDISP_TOGGLE_LOCK	33223
Others	ID_RELDISP_TOGGLE_LOCK_OTHER	33224
Selected Layer	ID_RELDISP_UNLOCK_LAYER	33225
Others	ID_RELDISP_UNLOCK_LAYER_OTHER	33226
Remove all Data Sources	ID_REMOVEALLDATASOURCES	36621
Remove	ID_REMOVE_DATASOURCE	36622
Remove from Favorites	ID_REMOVE_FAVORITE_ITEM	36623
Remove	ID_REMOVE_FROM_PROJECT	36624
All rows	ID_RETRIEVE_ALLROWS	36625
First n rows	ID_RETRIEVE_FIRSTROWS	36626
Save Project	ID_SAVEPROJECT	36627
Save Project As...	ID_SAVEPROJECTAS	36628

© 2015 Altova GmbH

4.5 Accessing AuthenticDesktopAPI

The focus of this documentation is the ActiveX controls and interfaces required to integrate the Authentic Desktop user interface into your application. To allow you to automate or control the functionality of the integrated components, the following properties give you access to the Authentic Desktop automation interface (AuthenticDesktopAPI):

`AuthenticDesktopControl.Application`
`AuthenticDesktopControlDocument.Document`
`AuthenticDesktopControlPlaceHolder.Project`

Some restrictions apply to the usage of the Authentic Desktop automation interface when integrating AuthenticDesktopControl at document-level. See Integration at document level for details.

4.6 Object Reference

Objects:
Authentic DesktopCommand
Authentic DesktopCommands
AuthenticDesktopControl
AuthenticDesktopControlDocument
AuthenticDesktopControlPlaceHolder

To give access to standard Authentic Desktop functionality, objects of the **Authentic Desktop automation interface** can be accessed as well. See AuthenticDesktopControl.Application, AuthenticDesktopControlDocument.Document and AuthenticDesktopControlPlaceHolder.Project for more information.

© 2015 Altova GmbH

4.6.1 Authentic DesktopCommand

Properties:
ID
Label
IsSeparator
ToolTip
StatusText
Accelerator
SubCommands

Description:
Each Command object can be one of three possible types:

- **Command**: ID is set to a value greater 0 and Label is set to the command name. IsSeparator is false and the SubCommands collection is empty.
- **Separator**: IsSeparator is true. ID is 0 and Label is not set. The SubCommands collection is empty.
- (**Sub**) **Menu**: The SubCommands collection contains Command objects and Label is the name of the menu. ID is set to 0 and IsSeparator is false.

Accelerator

Property: Label as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is the accelerator key defined for the command. If the command has no accelerator key assigned, this property returns the empty string.

The string representation of the accelerator key has the following format:

```
[ALT+][CTRL+][SHIFT+]key
```

Where key is converted using the Windows Platform SDK function GetKeyNameText.

ID

Property: ID as long

Description:
ID is 0 for separators and menus.
For commands, this is the ID which can be used with Exec and QueryStatus.

IsSeparator

Property: IsSeparator as boolean

Description:
True if the command is a separator.

Label

Property: `Label` as string

Description:
Label is empty for separators.
For command objects that are children of the ALL_COMMANDS collection, this is a unique name. Command icons are stored in files with this name. See Query Commands for more information.

For command objects that are children of menus, the label property holds the command's menu text.
For sub-menus, this property holds the menu text.

StatusText

Property: `Label` as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is the text shown in the status bar when the command is selected.

SubCommands

Property: `SubCommands` as `Commands`

Description:
The `SubCommands` collection holds any sub-commands if this command is actually a menu or submenu.

ToolTip

Property: `ToolTip` as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is the text shown as tool-tip.

© 2015 Altova GmbH

4.6.2 Authentic DesktopCommands

Properties:
Count
Item

Description:
Collection of Command objects to get access to command labels and IDs of the AuthenticDesktopControl. Those commands can be executed with the Exec method and their status can be queried with QueryStatus.

Count

Property: Count as long

Description:
Number of Command objects on this level of the collection.

Item

Property: Item (n as long) as Command

Description:
Gets the command with the index n in this collection. Index is 1-based.

4.6.3 AuthenticDesktopControl

Properties:
IntegrationLevel
Appearance
Application
BorderStyle
CommandsList
CommandsStructure (deprecated)
EnableUserPrompts
MainMenu
Toolbars

Methods:
Open
Exec
QueryStatus

Events:
OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged
OnDocumentOpened
OnValidationWindowUpdated

This object is a complete ActiveX control and should only be visible if the Authentic Desktop library is used in the Application Level mode.

Properties

The following properties are defined:

IntegrationLevel
EnableUserPrompts
Appearance
BorderStyle

Command related properties:
CommandsList
MainMenu
Toolbars
CommandsStructure (deprecated)

Access to AuthenticDesktopAPI:
Application

Appearance

Property: Appearance as short

Dispatch Id: *-520*

© 2015 Altova GmbH

Description:
A value not equal to `0` displays a client edge around the control. Default value is `0`.

Application

Property: `Application` as Application

Dispatch Id: *1*

Description:
The `Application` property gives access to the `Application` object of the complete Authentic Desktop automation server API. The property is read-only.

BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

CommandsList

Property: `CommandList` as `Commands` (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with AuthenticDesktopControl.

EnableUserPrompts

Property: `EnableUserPrompts` as boolean

Dispatch Id: *1006*

Description:
Setting this property to *false*, disables user prompts in the control. The default value is *true*.

IntegrationLevel

Property: `IntegrationLevel` as `ICActiveXIntegrationLevel`

Dispatch Id: *1000*

Description:
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at the application level and Integration at document level for more information.

Note: It is important to set this property immediately after the creation of the `AuthenticDesktopControl` object.

MainMenu

Property: `MainMenu` as `Command` (read-only)

Dispatch Id: *1003*

Description:
This property gives access to the description of the AuthenticDesktopControl main menu.

Toolbars

Property: `Toolbars` as `Commands` (read-only)

Dispatch Id: *1005*

Description:
This property returns a list of all toolbar descriptions that describe all toolbars available with AuthenticDesktopControl.

Methods

The following methods are defined:

`Open`
`Exec`
`QueryStatus`

Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *6*

Description:
`Exec` calls the `Authentic Desktop` command with the ID `nCmdID`. If the command can be executed, the method returns `true`. See also `CommandsStructure` to get a list of all available commands and `QueryStatus` to retrieve the status of any command.

© 2015 Altova GmbH

Open

Method: `Open` (`strFilePath` as string) as boolean

Dispatch Id: *5*

Description:
The result of the method depends on the extension passed in the argument `strFilePath`. If the file extension is `.sps`, a new document is opened. If the file extension is `.svp`, the corresponding project is opened. If a different file extension is passed into the method, the control tries to load the file as a new component into the active document.

Do not use this method to load documents or projects when using the control in document-level integration mode. Instead, use `AuthenticDesktopControlDocument.Open` and `AuthenticDesktopControlPlaceHolder.OpenProject`.

QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *7*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid Authentic Desktop command. If `QueryStatus` returns a value of `1` or `5`, the command is disabled.

Events

The AuthenticDesktopControl ActiveX control provides the following connection point events:

`OnUpdateCmdUI`
`OnOpenedOrFocused`
`OnCloseEditingWindow`
`OnFileChangedAlert`
`OnContextChanged`

`OnDocumentOpened`
`OnValidationWindowUpdated`

OnCloseEditingWindow

Event: `OnCloseEditingWindow` (i_strFilePath as String) as boolean

Dispatch Id: *1002*

Description:
This event is triggered when Authentic Desktop needs to close an already open document. As an answer to this event, clients should close the editor window associated with *i_strFilePath*. Returning *true* from this event indicates that the client has closed the document. Clients can return *false* if no specific handling is required and AuthenticDesktopControl should try to close the editor and destroy the associated document control.

OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in Authentic Desktop

OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1*

Description:
This event is triggered whenever a document is opened. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface and can be used to query for more details about the document, or perform additional operations. When integrating on document-level, it is often better to use the event `AuthenticDesktopControlDocument.OnDocumentOpened` instead.

OnFileChangedAlert

Event: `OnFileChangedAlert` (`i_strFilePath` as String) as bool

Dispatch Id: *1001*

Description:
This event is triggered when a file loaded with AuthenticDesktopControl, is changed on the harddisk by another application. Clients should return true, if they handled the event, or false, if Authentic Desktop should handle it in its customary way, i.e. prompting the user for reload.

OnLicenseProblem

Event: `OnLicenseProblem` (`i_strLicenseProblemText` as String)

Dispatch Id: *1005*

© 2015 Altova GmbH

Description:
This event is triggered when AuthenticDesktopControl detects that no valid license is available for this control. In case of restricted user licenses this can happen some time after the control has been initialized. Integrators should use this event to disable access to this control's functionality. After returning from this event, the control will block access to its functionality (e.g. show empty windows in its controls and return errors on requests).

OnOpenedOrFocused

Event: `OnOpenedOrFocused` (`i_strFilePath` as String, `i_bOpenWithThisControl` as bool)

Dispatch Id: *1000*

Description:
When integrating at application level, this event informs clients that a document has been opened, or made active by Authentic Desktop.

When integrating at document level, this event instructs the client to open the file `i_strFilePath` in a document window. If the file is already open, the corresponding document window should be made the active window.

if `i_bOpenWithThisControl` is `true`, the document must be opened with AuthenticDesktopControl, since internal access is required. Otherwise, the file can be opened with different editors.

OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` as long `)`

Dispatch Id: *1006*

Description:
This event is triggered when the tool window is updated.

OnUpdateCmdUI

Event: `OnUpdateCmdUI ()`

Dispatch Id: *1003*

Description:
Called frequently to give integrators a good opportunity to check status of Authentic Desktop commands using `AuthenticDesktopControl.QueryStatus`. Do not perform long operations in this callback.

OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window, is updated with new information.

© 2015 Altova GmbH

4.6.4 AuthenticDesktopControlDocument

Properties:
Appearance
BorderStyle
Document
IsModified
Path
ReadOnly

Methods:
Exec
New
Open
QueryStatus
Reload
Save
SaveAs

Events:
OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate

If the AuthenticDesktopControl is integrated in the Document Level mode each document is displayed in an own object of type `AuthenticDesktopControlDocument`. The `AuthenticDesktopControlDocument` contains only one document at the time but can be reused to display different files one after another.

This object is a complete ActiveX control.

Properties

The following properties are defined:

ReadOnly
IsModified
Path
Appearance
BorderStyle

Access to AuthenticDesktopAPI:
Document

Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:

A value not equal to `0` displays a client edge around the document control. Default value is `0`.

BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

Document

Property: `Document` as Document

Dispatch Id: *1*

Description:
The `Document` property gives access to the `Document` object of the Authentic Desktop automation server API. This interface provides additional functionalities which can be used with the document loaded in the control. The property is read-only.

IsModified

Property: `IsModified` as boolean (read-only)

Dispatch Id: *1006*

Description:
`IsModified` is *true* if the document content has changed since the last open, reload or save operation. It is *false*, otherwise.

Path

Property: `Path` as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

ReadOnly

Property: `ReadOnly` as boolean

Dispatch Id: *1007*

© 2015 Altova GmbH

Description:
Using this property you can turn on and off the read-only mode of the document. If `ReadOnly` is `true` it is not possible to do any modifications.

Methods

The following methods are defined:

Document handling:
`New`
`Open`
`Reload`
`Save`
`SaveAs`

Command Handling:
`Exec`
`QueryStatus`

Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *8*

Description:
`Exec` calls the Authentic Desktop command with the ID `nCmdID`. If the command can be executed, the method returns `true`. The client should call the `Exec` method of the document control if there is currently an active document available in the application.

See also `CommandsStructure` to get a list of all available commands and `QueryStatus` to retrieve the status of any command.

New

Method: `New` () as boolean

Dispatch Id: *1000*

Description:
This method initializes a new document inside the control..

Open

Method: `Open` (`strFileName` as string) as boolean

Dispatch Id: *1001*

Description:

Open loads the file strFileName as the new document into the control.

QueryStatus

Method: QueryStatus (nCmdID as long) as long

Dispatch Id: *9*

Description:
QueryStatus returns the enabled/disabled and checked/unchecked status of the command specified by nCmdID. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if QueryStatus returns 0 the command ID is not recognized as a valid Authentic Desktop command. If QueryStatus returns a value of 1 or 5 the command is disabled. The client should call the QueryStatus method of the document control if there is currently an active document available in the application.

Reload

Method: Reload () as boolean

Dispatch Id: *1002*

Description:
Reload updates the document content from the file system.

Save

Method: Save () as boolean

Dispatch Id: *1003*

Description:
Save saves the current document at the location Path.

SaveAs

Method: SaveAs (strFileName as string) as boolean

Dispatch Id: *1004*

Description:
SaveAs sets Path to *strFileName* and then saves the document to this location.

© 2015 Altova GmbH

Events

The AuthenticDesktopControlDocument ActiveX control provides following connection point events:

OnDocumentOpened
OnDocumentClosed
OnModifiedFlagChanged
OnContextChanged
OnFileChangedAlert
OnActivate
OnSetEditorTitle

OnActivate

Event: `OnActivate` ()

Dispatch Id: *1005*

Description:
This event is triggered when the document control is activated, has the focus, and is ready for user input.

OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:

OnDocumentClosed

Event: `OnDocumentClosed` (`objDocument` as Document)

Dispatch Id: *1001*

Description:
This event is triggered whenever the document loaded into this control is closed. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface and should be used with care.

OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1000*

Description:
This event is triggered whenever a document is opened in this control. The argument `objDocument` is a `Document` object from the Authentic Desktop automation interface, and can

be used to query for more details about the document, or perform additional operations.

OnDocumentSaveAs

Event: `OnContextDocumentSaveAs` (`i_strFileName` as String)

Dispatch Id: *1007*

Description:
This event is triggered when this document gets internally saved under a new name.

OnFileChangedAlert

Event: `OnFileChangedAlert` () as bool

Dispatch Id: *1003*

Description:
This event is triggered when the file loaded into this document control, is changed on the harddisk by another application. Clients should return true, if they handled the event, or false, if Authentic Desktop should handle it in its customary way, i.e. prompting the user for reload.

OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1002*

Description:
This event gets triggered whenever the document changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the document contents differs from the original content, and *false*, otherwise.

OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

© 2015 Altova GmbH

4.6.5 AuthenticDesktopControlPlaceHolder

Properties available for all kinds of placeholder windows:
PlaceholderWindowID

Properties for project placeholder window:
Project

Methods for project placeholder window:
OpenProject
CloseProject

The AuthenticDesktopControlPlaceHolder control is used to show the additional Authentic Desktop windows like Overview, Library or Project window. It is used like any other ActiveX control and can be placed anywhere in the client application.

Properties

The following properties are defined:

PlaceholderWindowID

Access to AuthenticDesktopAPI:
Project

Label

Property: Label as String (read-only)

Dispatch Id: *1001*

Description:
This property gives access to the title of the placeholder. The property is read-only.

PlaceholderWindowID

Property: PlaceholderWindowID as

Dispatch Id: *1*

Description:
Using this property the object knows which Authentic Desktop window should be displayed in the client area of the control. The PlaceholderWindowID can be set at any time to any valid value of the enumeration. The control changes its state immediately and shows the new Authentic Desktop window.

Project

Property: `Project` as Project (read-only)

Dispatch Id: *2*

Description:
The `Project` property gives access to the `Project` object of the Authentic Desktop automation server API. This interface provides additional functionalities which can be used with the project loaded into the control. The property will return a valid project interface only if the placeholder window has `PlaceholderWindowID` with a value of `Authentic DesktopXProjectWindow (=3)`. The property is read-only.

Methods

The following method is defined:

`OpenProject`
`CloseProject`

OpenProject

Method: `OpenProject` (`strFileName` as string) as boolean

Dispatch Id: *3*

Description:
`OpenProject` loads the file `strFileName` as the new project into the control. The method will fail if the placeholder window has a `PlaceholderWindowID` different to `XMLSpyXProjectWindow (=3)`.

CloseProject

Method: `CloseProject` ()

Dispatch Id: *4*

Description:
`CloseProject` closes the project loaded the control. The method will fail if the placeholder window has a `PlaceholderWindowID` different to `Authentic DesktopXProjectWindow (=3)`.

Events

The AuthenticDesktopControlPlaceholder ActiveX control provides following connection point events:

`OnModifiedFlagChanged`

© 2015 Altova GmbH

OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1*

Description:
This event gets triggered only for placeholder controls with a `PlaceholderWindowID` of `Authentic DesktopXProjectWindow (=3)`. The event is fired whenever the project content changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the project contents differs from the original content, and *false*, otherwise.

OnSetLabel

Event: `OnSetLabel(i_strNewLabel` as string`)`

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

4.6.6 Enumerations

The following enumerations are defined:

ICActiveXIntegrationLevel
AuthenticDesktopControlPlaceholderWindow

ICActiveXIntegrationLevel

Possible values for the IntegrationLevel property of the AuthenticDesktopControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

AuthenticDesktopControlPlaceholderWindow

This enumeration contains the list of the supported additional Authentic Desktop windows.

```
AuthenticDesktopControlNoToolWnd                    =  -1
AuthenticDesktopControlEntryHelperTopToolWnd        =   0
AuthenticDesktopControlEntryHelperMiddleToolWnd     =   1
AuthenticDesktopControlEntryHelperBottomToolWnd     =   2
AuthenticDesktopControlValidatorOutputToolWnd       =   3
AuthenticDesktopControlProjectWindowToolWnd         =   4
AuthenticDesktopControlXSLTDebuggerContextToolWnd   =   5
AuthenticDesktopControlXSLTDebuggerCallstackToolWnd =   6
AuthenticDesktopControlXSLTDebuggerVariableToolWnd  =   7
AuthenticDesktopControlXSLTDebuggerWatchToolWnd     =   8
AuthenticDesktopControlXSLTDebuggerTemplateToolWnd  =   9
AuthenticDesktopControlXSLTDebuggerInfoToolWnd      =  10
AuthenticDesktopControlXSLTDebuggerMessageToolWnd   =  11
AuthenticDesktopControlXSLTDebuggerTraceToolWnd     =  12
AuthenticDesktopControlSOAPDebuggerToolWnd          =  13
AuthenticDesktopControlXPathProfilerListToolWnd     =  14
AuthenticDesktopControlXPathProfilerTreeToolWnd     =  15
AuthenticDesktopControlXPathDialogToolWnd           =  16
AuthenticDesktopControlDBQueryManagerToolWnd        =  17
AuthenticDesktopControlInfoToolWnd                  =  18
AuthenticDesktopControlXSLOutlineToolWnd            =  19
AuthenticDesktopControlSchemaFindToolWnd            =  20
```


© 2015 Altova GmbH

Altova Authentic 2016 Community Edition

Anexos

Anexos

Estos anexos contienen información técnica sobre Authentic Desktop e información importante sobre las licencias. Cada anexo contiene varios apartados.

Datos técnicos

- Requisitos de sistema operativo y de memoria
- Analizador XML de Altova
- Motores XSLT y XQuery de Altova
- Compatiblidad con Unicode
- Uso de internet

Información sobre licencias

- Distribución electrónica del software
- Derechos de autor
- Contrato de licencia para el usuario final

© 2015 Altova GmbH

1 Datos técnicos

Esta sección incluye información general práctica sobre aspectos técnicos de su software. Consta de varios apartados:

- Requisitos de OS y memoria
- Analizador XML de Altova
- Motores XSLT y XQuery de Altova
- Compatibilidad con Unicode
- Uso de Internet

1.1 Requisitos de SO y memoria

Sistema operativo
Las aplicaciones de software de Altova están disponibles en estas plataformas:

- Aplicaciones de 32 bits para Windows XP, Windows Vista, Windows 7, Windows 8, Windows Server 2003 y 2008
- Aplicaciones de 64 bits para Windows Vista, Windows 7, Windows 8 y Windows Server 2012

Memoria
Puesto que el software está escrito en C++ no necesita tanto espacio como un JRE y suele necesitar menos memoria que otras aplicaciones similares basadas en Java. No obstante, todos los documentos se cargan en memoria por completo, para poder analizarlos completamente y mejorar la velocidad de visualización y edición. Los requisitos de memoria aumentan en función del tamaño del documento.

Los requisitos de memoria también vienen dados por el historial de operaciones Deshacer. Cuando se cortan y pegan secciones grandes de documentos de gran tamaño, la memoria disponible se puede agotar rápidamente.

© 2015 Altova GmbH

1.2 Validador XML de Altova

Al abrir un documento XML, la aplicación usa su validador integrado para comprobar si el formato XML es correcto, validar el documento con un esquema (si viene dado) y generar árboles e infosets. El validador XML también se usa para ofrecerle ayuda de edición inteligente mientras edita documentos y para mostrar de forma dinámica los errores de validación (si los hay).

El validador XML implementa la recomendación final de la especificación XML Schema 1.0 y 1.1 del W3C. Además Altova actualiza continuamente el validador los avances del grupo de trabajo XML Schema del W3C.

1.3 Motores XSLT y XQuery de Altova

Los productos de Altova usan los motores XSLT 1.0, XSLT 2.0, XSLT 3.0, XQuery 1.0 y XQuery 3.0 de Altova. Para más información sobre el comportamiento de cada motor, consulte el anexo *Información sobre motores de Altova.*

Nota: MapForce de Altova genera un código que usa los motores XSLT 1.0, 2.0 y XQuery 1.0.

© 2015 Altova GmbH

1.4 Compatibilidad con Unicode

Los productos XML de Altova son completamente compatibles con Unicode. Para editar un documento XML también necesitará una fuente compatible con los caracteres Unicode utilizados por el documento.

Tenga en cuenta que la mayoría de las fuentes contienen solamente un subconjunto muy concreto de caracteres Unicode y, por tanto, están destinadas a un sistema de escritura concreto. Si algunos caracteres aparecen desfigurados, el motivo puede ser que la fuente seleccionada no contiene los glifos necesarios. Por tanto, es recomendable tener una fuente que abarque todos los caracteres Unicode. Sobre todo si edita documentos XML en varios idiomas o sistemas de escritura. Una fuente Unicode que suele venir con los equipos Windows es la fuente Arial Unicode MS.

En la carpeta `/Examples` de la carpeta de su aplicación puede encontrar un archivo XHTML llamado `UnicodeUTF-8.html` que incluye esta frase en gran número de idiomas y sistemas de escritura diferentes:

- *When the world wants to talk, it speaks Unicode*
- *Cuando el mundo quiere conversar, habla Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。

Abra este archivo XHTML y observe el potencial de Unicode.

1.5 Uso de Internet

Las aplicaciones de Altova inician conexiones a Internet en estos casos:

- Si hace clic en el botón **Solicitar una clave de evaluación GRATUITA** del cuadro de diálogo "Activación del software" (**Ayuda | Activación del software**), los campos del cuadro de diálogo de activación del software se transfieren a nuestro servidor web por medio de una conexión HTTP corriente (puerto 80) y le enviamos el código de evaluación gratuito por correo electrónico.
- En algunos productos de Altova puede abrir un archivo por Internet (**Archivo | Abrir | Cambiar a URL**). En este caso, el documento se recupera usando uno de estos protocolos y conexiones: HTTP (normalmente por el puerto 80), FTP (normalmente por el puerto 20/21) o HTTPS (normalmente por el puerto 443). También puede ejecutar un servidor HTTP en el puerto 8080. (En el cuadro de diálogo "Abrir URL", después del nombre de servidor escriba dos puntos y el número de puerto.)
- Si abre un documento XML que hace referencia a un documento DTD o esquema XML y el documento se especifica a través de una URL, el documento de esquema al que se hace referencia también se recupera a través de una conexión HTTP (puerto 80) o cualquier otro protocolo (ver punto anterior). El documento de esquema también se recupera para validar el archivo XML. Recuerde que la validación puede realizarse automáticamente nada más abrir el documento, si seleccionó esta opción en la pestaña *Archivo* del cuadro de diálogo Opciones (**Herramientas | Opciones**).
- En las aplicaciones de Altova que trabajen con WSDL y SOAP, las conexiones a servicios web son definidas por documentos WSDL.
- Si usa el comando **Archivo | Enviar por correo electrónico** en XMLSpy, el texto seleccionado actualmente o el archivo se envía con el programa de correo electrónico instalado en el equipo.
- Durante la activación del software y la búsqueda de actualizaciones, tal y como se describe en el contrato de licencia de software de Altova.

© 2015 Altova GmbH

2 Información sobre licencias

Esta sección incluye:

- información sobre la distribución de este producto de software.
- información sobre los derechos de propiedad intelectual relacionados con este producto de software.
- el contrato de licencia para el usuario final de Authentic que rige el uso de este producto de software

Los términos del contrato de licencia que aceptó al instalar el producto de software son vinculantes, por lo que rogamos lea atentamente toda esta información.

2.1 Distribución electrónica de software

Este producto está disponible por distribución electrónica de software, un método de distribución que ofrece ventajas únicas:

- Puede evaluar el software de forma totalmente gratuita antes de decidir si compra el producto.
- Si decide comprarlo, puede hacer un pedido en línea en el sitio web de Altova y conseguir en pocos minutos el software con licencia.
- Si realiza el pedido en línea, siempre recibirá la versión más reciente de nuestro software.
- El paquete de instalación del producto incluye un sistema de ayuda en pantalla totalmente integrado. La versión más reciente del manual del usuario está disponible en www.altova.com (i) en formato HTML y (ii) en formato PDF para descargar e imprimir si lo desea.

Período de evaluación de 30 días
Después de descargar el producto de software, puede probarlo de forma totalmente gratuita durante un plazo de 30 días. Pasados unos 20 días, el software empieza a recordarle que no tiene una licencia. El mensaje de aviso aparece una sola vez al iniciarse la aplicación. Si desea usar el programa una vez pasado el plazo de 30 días, deberá comprar una licencia permanente y aceptar el contrato de licencia de software de Altova, que se entrega en forma de código clave. La licencia puede comprarse directamente en la tienda en línea del sitio web de Altova. Después de comprar la licencia recibirá el código clave, que debe introducir en el cuadro de diálogo "Activación del software" para desbloquear el producto de forma permanente.

Distribuir la versión de evaluación a otros usuarios de su organización
Si desea distribuir la versión de evaluación en la red de su compañía o si desea usarlo en un PC que no está conectado a Internet, solamente puede distribuir los programas de instalación (siempre y cuando no se modifiquen de forma alguna). Todo usuario que acceda al instalador debe solicitar su propio código clave de evaluación (de 30 días). Una vez pasado este plazo de 30 días, todos los usuarios deben comprar también una licencia para poder seguir usando el producto.

Para más información consulte el contrato de licencia de software de Altova que aparece al final de esta sección.

© 2015 Altova GmbH

2.2 Derechos de propiedad intelectual

El software de Altova y sus copias (si tiene permiso de Altova para realizar copias) es propiedad intelectual de Altova y de sus proveedores. La estructura, la organización y el código del software se considera secreto comercial e información confidencial de Altova y de sus proveedores. El software está protegido por las leyes de derechos de autor, como la ley de derechos de autor de EE UU, tratados internacionales y la legislación vigente del país donde se utiliza, entre otras. Altova conserva los derechos de propiedad de todas las patentes, derechos de autor, secretos comerciales, marcas registradas y otros derechos de propiedad intelectual pertenecientes al software y los derechos de propiedad de Altova abarcan también imágenes, fotografías, animaciones, vídeos, audio, música, texto y otros applets incorporados al software y al material impreso que viene con el software. Las notificaciones de infracción de dichos derechos de autor debe enviarse al agente de derechos de autor de Altova, cuyos datos de contacto aparecen en el sitio web de Altova.

El software de Altova contiene software de terceros que también está protegido por las leyes de propiedad intelectual, incluida, entre otras, la legislación de derechos de autor mencionada en http://www.altova.com/es/legal_3rdparty.html.

Los demás nombres y marcas registradas son propiedad de sus respectivos propietarios.

2.3 Altova Authentic: contrato de licencia para el usuario final

Altova End-User License Agreement for Authentic
THIS IS A LEGAL DOCUMENT -- RETAIN FOR YOUR RECORDS
ALTOVA® END-USER LICENSE AGREEMENT
For Authentic® Enterprise Software Editions
And Authentic® Community Software Editions

Licensor:
Altova GmbH
Rudolfsplatz 13a/9
A-1010 Wien
Austria

Important - Read Carefully. Notice to User:

This Altova End User License Agreement for Authentic® ("AEULA") governs your right to (i) use the Authentic Desktop Enterprise Edition software, (ii) use, reproduce and distribute the Authentic Browser-Plugin Enterprise Edition software, (iii) use the Authentic Desktop Community Edition software and (iv) use, reproduce and distribute the Authentic Browser-Plugin Community Edition software (each or collectively hereinafter referenced, as "Authentic Software"). Your license rights depend on the specific software edition that you have licensed as some editions have different rights and restrictions applicable to them as set forth in detail below. This Authentic EULA is a legal document between you and Altova GmbH ("Altova"). It is important that you read this document before using the Altova-provided software and any accompanying documentation, including, without limitation, printed materials, 'online' files, or electronic documentation ("Documentation"). By clicking the "I accept" and "Next" buttons below, or by installing, or otherwise using the Authentic Software, you agree to be bound by the terms of this AEULA as well as the Altova Privacy Policy ("Privacy Policy") including, without limitation, the warranty disclaimers, limitation of liability, data use and termination provisions below. You agree that this agreement is enforceable like any written agreement negotiated and signed by you. If you do not agree, you are not licensed to use the Authentic Software, and you must destroy any downloaded copies of the Authentic Software in your possession or control. Please go to our Web site at http://www.altova.com/authenticeula to download and print a copy of this Authentic EULA for your files and http://www.altova.com/privacy to review the Privacy Policy.

1. Authentic Desktop Enterprise Edition ("ADEE") Software Terms and Conditions

(a) License Grant. Upon your acceptance of this AEULA, Altova grants you a non-exclusive, non-transferable (except as provided below), limited license, without the right to grant sublicenses, to install and use a copy of ADEE software on one compatible personal computer or workstation up to the Permitted Number of computers. You may not distribute or reproduce the ADEE software other than as expressly permitted. Subject to the limitations set forth in Section 1(a)(i) you may install and use a copy of this software on more than one of your compatible personal computers or workstations if you have purchased a Named-User license. Subject to the limitations set forth in Sections 1(a)(iii) and 1(a)(iv), users may use the software concurrently on a network. The Permitted Number of computers and/or users and the type of license, e.g. Installed, Named-User, and Concurrent-User, shall be determined and specified at such time as you elect to purchase the software. Installed user licenses are intended to be fixed and not concurrent. In other words, you cannot uninstall for one user in order to reinstall that license to a different user and then uninstall and reinstall back to the original user. Users should be static. Notwithstanding

the foregoing, permanent switchovers are acceptable (i.e., an employee has left the company, machine is retired). During the evaluation period, hereinafter defined, only a single user may install and use the ADEE software on one (1) personal computer or workstation. You may install one (1) copy of the ADEE software on a computer file server within your internal network solely for the purpose of downloading and installing this ADEE software onto other computers within your internal network up to the Permitted Number of computers in a commercial environment only. No other network use is permitted, including without limitation using the ADEE software either directly or through commands, data or instructions from or to a computer not part of your internal network, for Internet or Web-hosting services or by any user not licensed to use this copy of ADEE software through a valid license from Altova, except as set forth in Sections 1(a)(iii) and (iv) below Altova makes no warranties or representations about the performance of Altova software in a terminal server environment and the foregoing are expressly excluded from the limited warranty in Section 3(c) hereof and technical support is not available with respect to issues arising from use in such an environment.

(i) If you have licensed the "Named-User" version of the ADEE software, you may install the software on up to five (5) compatible personal computers or workstations of which you are the primary user thereby allowing you to switch from one computer to the other as necessary provided that only one (1) instance of the ADEE software will be used by you as the Named-User at any given time. If you have purchased multiple Named-User licenses, each individual Named-User will receive a separate license key code.

(ii) If you have licensed a "Concurrent-User" version of the ADEE software, you may install the ADEE software on any compatible computers in a commercial environment only, up to ten (10) times the Permitted Number of users, provided that only the Permitted Number of users actually use the software at the same time and further provided that the computers on which the ADEE software is installed are on the same physical computer network. The Permitted Number of concurrent users shall be delineated at such time as you elect to purchase the ADEE licenses. Each separate physical network or office location requires its own set of separate Concurrent User Licenses for those wishing to use the Concurrent-User versions of the software in more than one location or on more than one network, all subject to the above Permitted Number limitations and based on the number of users using or needing access to the software. If a computer is not on the same physical network, then a locally installed user license or a license dedicated to concurrent use in a virtual environment is required.

(iii) If you have licensed a "Concurrent-User" versions of ADEE software, you may install a copy of the ADEE Software on a terminal server (Microsoft Terminal Server, Citrix Metaframe, etc.), application virtualization server (Microsoft App-V, Citrix XenApp, VMWare ThinApp, etc.) or virtual machine environment within your internal network for the sole and exclusive purpose of permitting individual users within your organization to access and use the Software through a terminal server, application virtualization session, or virtual machine environment from another computer provided that the total number of users that access or use the ADEE Software concurrently at any given point in time on such network, virtual machine or terminal server does not exceed the Permitted Number; and provided that the total number of users authorized to use the ADEE Software through the terminal server, application virtualization session, or virtual machine environment does not exceed six (6) times the Permitted Number of users. Accordingly, the limitations set forth in Section 1(a)(ii) regarding the number of installations and the requirement that the usage be on the same physical network shall not apply to terminal server, application virtualization session, or virtual machine environments. In a virtual environment, you must deploy a means of preventing users from exceeding the Permitted Number of concurrent users. Altova makes no warranties or representations about the performance of Altova software in a terminal server, application virtualization session, or virtual machine

environment and the foregoing are expressly excluded from the limited warranty in Section 3 hereof and technical support is not available with respect to issues arising from use in such environments.

(iv) You may make one (1) backup and one (1) archival copy of the ADEE software, provided your backup and archival copies are not installed or used on any computer and further provided that all such copies shall bear the original and unmodified copyright, patent and other intellectual property markings that appear on or in the ADEE software. You may not transfer the rights to a backup or archival copy unless you transfer all rights in the ADEE software as provided in this AEULA. You, as the primary user of the computer on which the ADEE software is installed, may also install the ADEE software on one of your home computers for your use. A copy of the ADEE software may be installed on home computers up to a total of the number of Permitted Users provided that the ADEE software will not be used at the same time on a home computer as the ADEE software is being used on the primary computer. If you are using a Concurrent-User version of the ADEE software for home use, then you may install the software on any compatible computers equal to the number of Permitted Users only.

(b) Key Codes. Prior to your purchase and as part of the registration for the thirty (30) -day evaluation period, as applicable, you will receive an evaluation key code. You will receive a purchase key code when you elect to purchase the ADEE software licenses from either Altova GmbH or an authorized reseller. The purchase key code will enable you to activate the software beyond the initial evaluation period. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova.

(c) Limited Transfer Rights. You may transfer all your rights to use the ADEE software to another person or legal entity provided that: (i) you also transfer each of this AEULA, the ADEE software and all other software or hardware bundled or pre-installed with the ADEE software, including all copies, updates and prior versions, and all copies of font software converted into other formats, to such person or entity; (ii) you retain no copies, including backups and copies stored on a computer; (iii) the receiving party secures a personalized key code from Altova; and (iv) the receiving party accepts the terms and conditions of this AEULA and any other terms and conditions upon which you legally purchased a license to the ADEE software.

(d) Applicable AEULA Terms. The terms and conditions set forth in Sections 1, 3 and 7 apply to the ADEE software.

2. Authentic Browser-Plugin Enterprise Edition ("ABEE") Software Terms and Conditions

(a) License Grant and Term. Upon your acceptance of this AEULA, Altova grants you a non-exclusive, non-transferable limited license, without the right to grant sublicenses, to install and use ABEE software on a per server basis for a twelve (12) month term, commencing on the date of your license purchase and expiring on the date that is twelve (12) months thereafter (the "ABEE License Term"). Altova also grants you a non-exclusive, non-transferable, limited worldwide license, without the right to grant sublicenses, to use software to develop web pages, web applications, or applications that include ABEE software, to reproduce the ABEE software on your website or server and to distribute the ABEE software from your website or server over a computer network, but only in its executable object code form, and only to end users for the limited purpose of enabling them to view, share, and/or edit XML files during the ABEE License Term. If you wish to continue to use, and/or reproduce and/or distribute the ABEE software after the expiration of its license term, you must purchase a new Authentic Browser-Plugin Enterprise Edition. If you have purchased an ABEE software license then under the terms of the AEULA, support and maintenance (or SMP as further detailed below) for the software is included as part of the license purchase and you will be entitled to receive the benefits set forth below during the ABEE License

 © 2015 Altova GmbH

Term which is coterminous with the Support Period. Unlike other Altova software products, you **cannot** renew SMP for the ABEE software and at the expiration of the ABEE License Term and Support Period, you must purchase a new ABEE software license if you wish to continue to use, reproduce or distribute the ABEE software.

(b) Key Codes. Prior to your purchase you may request a thirty (30) day evaluation key code, which will be sent to you. **You will receive a purchase key code when you elect to purchase the ABEE software licenses from either Altova GmbH or an authorized reseller. The purchase key code will enable you to use the ABEE software during the ABEE License Term. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova.**

(c) ABEE Software Specific Restrictions. In addition to the restrictions and obligations provided in other sections of this AEULA that are applicable to the ABEE software, your limited license to distribute the ABEE software set forth above, is further subject to all of the following restrictions; (i) ABEE software may only be licensed but may not be sold, and (ii) you must use, reproduce or distribute the ABEE software provided by Altova **AS IS** and may not impair, alter or remove Altova's AEULA, (which will appear in the installation process and which an end user must accept in order to be able to install or operate the software or any other files).

(d) Applicable AEULA Terms. The terms and conditions set forth in Sections 2, 3 and 7 apply to the ABEE software.

3. Authentic Enterprise Editions (ADEE and ABEE) Software Additional Terms and Conditions

The terms set forth in Section 3 are applicable to the ADEE and ABEE software licenses and are in addition to the specific terms applicable to those software licenses.

(a) Upgrades and Updates. If the software that you have licensed is an upgrade or an update, then the latest update or upgrade that you download and install replaces all or part of the ADEE or ABEE software previously licensed The update or upgrade and the associated license keys, as applicable, does not constitute the granting of a second license to the software in that you may not use the upgrade or updated copy in addition to the copy of the software that it is replacing and whose license has terminated.

(b) Support and Maintenance. Altova offers "Support & Maintenance Package(s)" ("SMP") for the ADEE and ABEE software product editions that you have licensed. The Support Period, hereinafter defined, covered by such SMP shall be delineated at such time as you elect to purchase a SMP. In the case of your ABEE software license, twelve months of SMP is included that is coterminous with the ABEE License Term. Your rights with respect to support and maintenance as well as your upgrade eligibility depend on your decision to purchase SMP and the level of SMP that you have purchased:

(i) If you have not purchased SMP, you will receive the software AS IS and will not receive any maintenance releases or updates. However; Altova, at its option and in its sole discretion on a case by case basis, may decide to offer maintenance releases to you as a courtesy, but these maintenance releases will not include any new features in excess of the feature set at the time of your purchase of the Software. In addition, Altova will provide free technical support to you for thirty (30) days after the date of your purchase (the "Support Period" for the purposes of this paragraph 3(b)), and Altova, in its sole discretion on a case by case basis, may also provide free courtesy technical support during your thirty (30) -day evaluation period. Technical support is provided via a Web-based support form only, and there is no guaranteed

response time.

(ii) If you have purchased SMP then, solely for the duration of its delineated Support Period, **you are eligible to receive the version of the ADEE or ABEE software edition** that you have licensed and all maintenance releases and updates for that edition that are released during your Support Period. For the duration of your SMP's Support Period, you will also be eligible to receive upgrades to the comparable edition of the next version of the ADEE or ABEE software that succeeds the software edition that you have licensed for applicable upgrades released during your Support Period. The specific upgrade edition that you are eligible to receive based on your Support Period is further detailed in the SMP that you have purchased. Software that is introduced as a separate product is not included in SMP. Maintenance releases, updates and upgrades may or may not include additional features. In addition, Altova will provide Priority Technical Support to you for the duration of the Support Period. Priority Technical Support is provided via a Web-based support form only and Altova will make commercially reasonable efforts to respond via e-mail to all requests within forty-eight (48) hours during Altova's business hours (MO-FR, 8am UTC – 10pm UTC, Austrian and US holidays excluded) and to make reasonable efforts to provide work-arounds to errors reported in the software.

(iii) During the Support Period you may also report any software problem or error to Altova. If Altova determines that a reported reproducible material error in the software exists and significantly impairs the usability and utility of the ADEE or ABEE software, Altova agrees to use reasonable commercial efforts to correct or provide a usable work-around solution in an upcoming maintenance release or update, which is made available at certain times at Altova's sole discretion. If Altova, in its discretion, requests written verification of an error or malfunction discovered by you or requests supporting example files that exhibit the software problem, you shall promptly provide such verification or files, by email, telecopy, or overnight mail, setting forth in reasonable detail the respects in which the ADEE or ABEE software fails to perform. You shall use reasonable efforts to cooperate in diagnosis or study of errors. Altova may include error corrections in maintenance releases, updates, or new major releases of the software. Altova is not obligated to fix errors that are immaterial. Immaterial errors are those that do not significantly impact use of the software as determined by Altova in its sole discretion. Whether or not you have purchased the Support & Maintenance Package, technical support only covers issues or questions resulting directly out of the operation of the ADEE or ABEE software and Altova will not provide you with generic consultation, assistance, or advice under any circumstances.

(iv) Updating the ADEE or ABEE software may require the updating of software not covered by this AEULA before installation. Updates of the operating system and application software not specifically covered by this AEULA are your responsibility and will not be provided by Altova under this AEULA. Altova's obligations under this Section are contingent upon your proper use of the ADEE or ABEE software and your compliance with the terms and conditions of this AEULA at all times. Altova shall be under no obligation to provide the above technical support if, in Altova's opinion, the ADEE or ABEE software has failed due to the following conditions: (a) damage caused by the relocation of the software to another location or CPU; (b) alterations, modifications or attempts to change the software without Altova's written approval; (c) causes external to the software, such as natural disasters, the failure or fluctuation of electrical power, or computer equipment failure; (d) your failure to maintain the software at Altova's specified release level; or (e) use of the software with other software without Altova's prior written approval. It will be your sole responsibility to: (a) comply with all Altova-specified operating and troubleshooting procedures and then notify Altova immediately of ADEE or ABEE software malfunction and provide Altova with complete information thereof; (b) provide for the security of your confidential information; and (c) establish and maintain backup systems and procedures necessary to reconstruct lost or altered files, data or programs.

 © 2015 Altova GmbH

(c) Limited Warranty. Altova warrants to the person or entity that first purchases a license for use of the ADEE or ABEE software pursuant to the terms of this AEULA that (i) the software will perform substantially in accordance with any accompanying documentation for a period of ninety (90) days from the date of receipt, and (ii) any support services provided by Altova shall be substantially as described in Section 3(b) of this AEULA. Some states and jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you. To the extent allowed by applicable law, implied warranties on the ADEE or ABEE software, if any, are limited to ninety (90) days. Altova's and its suppliers' entire liability and your exclusive remedy shall be, at Altova's option, either (i) return of the price paid, if any, or (ii) repair or replacement of the ADEE or ABEE software that does not meet Altova's Limited Warranty and which is returned to Altova with a copy of your receipt. This Limited Warranty is void if failure of the ADEE or ABEE software has resulted from accident, abuse, misapplication, abnormal use, Trojan horse, virus, or any other malicious external code. Any replacement ADEE or ABEE software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. This limited warranty does not apply to Evaluation Software. THE FOREGOING LIMITED WARRANTY AND REMEDIES STATE THE SOLE AND EXCLUSIVE REMEDIES FOR ALTOVA'S OR ITS SUPPLIERS' BREACH OF WARRANTY. ALTOVA AND ITS SUPPLIERS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, AND FOR ANY WARRANTY, CONDITION, REPRESENTATION OR TERM TO THE EXTENT WHICH THE SAME CANNOT OR MAY NOT BE EXCLUDED OR LIMITED BY LAW APPLICABLE TO YOU IN YOUR JURISDICTION, ALTOVA AND ITS SUPPLIERS MAKE NO WARRANTIES, CONDITIONS, REPRESENTATIONS OR TERMS, EXPRESS OR IMPLIED, WHETHER BY STATUTE, COMMON LAW, CUSTOM, USAGE OR OTHERWISE AS TO ANY OTHER MATTERS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, ALTOVA AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, INFORMATIONAL CONTENT OR ACCURACY, QUIET ENJOYMENT, TITLE AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION.

(d) Limitation of Liability and Infringement Claims. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW EVEN IF A REMEDY FAILS ITS ESSENTIAL PURPOSE, IN NO EVENT SHALL ALTOVA OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE ADEE or ABEE SOFTWARE OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF ALTOVA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, ALTOVA'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AEULA SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE ADEE OR ABEE SOFTWARE PRODUCT. Because some states and jurisdictions do not allow the exclusion or limitation of liability, the above limitation may not apply to you. In such states and jurisdictions, Altova's liability shall be limited to the greatest extent permitted by law and the limitations or exclusions of warranties and liability contained herein do not prejudice applicable statutory consumer rights of person acquiring goods otherwise than in the course of business. The disclaimer and limited liability above are fundamental to this AEULA between Altova and you. Altova will indemnify and hold you harmless and will defend or settle any claim, suit or proceeding brought against you by a third party that is based upon a claim that the content contained in the

ADEE or ABEE software infringes a copyright or violates an intellectual or proprietary right protected by United States or European Union law ("Claim"), but only to the extent the Claim arises directly out of the use of the Software and subject to the limitations set forth in this Section 3(d) of this Agreement except as otherwise expressly provided. You must notify Altova in writing of any Claim within ten (10) business days after you first receive notice of the Claim, and you shall provide to Altova at no cost such assistance and cooperation as Altova may reasonably request from time to time in connection with the defense of the Claim. Altova shall have sole control over any Claim (including, without limitation, the selection of counsel and the right to settle on your behalf on any terms Altova deems desirable in the sole exercise of its discretion). You may, at your sole cost, retain separate counsel and participate in the defense or settlement negotiations. Altova shall pay actual damages, costs, and attorney fees awarded against you (or payable by you pursuant to a settlement agreement) in connection with a Claim to the extent such direct damages and costs are not reimbursed to you by insurance or a third party, to an aggregate maximum equal to the purchase price of the ADEE or ABEE software. If the ADEE or ABEE software or its use becomes the subject of a Claim or its use is enjoined, or if in the opinion of Altova's legal counsel the software is likely to become the subject of a Claim, Altova shall attempt to resolve the Claim by using commercially reasonable efforts to modify the ADEE or ABEE software or obtain a license to continue using the ADEE or ABEE software. If in the opinion of Altova's legal counsel the Claim, the injunction or potential Claim cannot be resolved through reasonable modification or licensing, Altova, at its own election, may terminate this AEULA without penalty, and will refund to you on a pro rata basis any fees paid in advance by you to Altova. THE FOREGOING CONSTITUTES ALTOVA'S SOLE AND EXCLUSIVE LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT. This indemnity does not apply to infringements that would not be such, except for customer-supplied elements.

(e) Evaluation Software. This section applies to all evaluation copies of the ADEE or ABEE software ("Evaluation Software") and continues in effect until you purchase a license. THE EVALUATION SOFTWARE IS PROVIDED TO YOU "AS-IS" WITH NO WARRANTIES FOR USE OR PERFORMANCE, AND ALTOVA DISCLAIMS ANY WARRANTY OR LIABILITY OBLIGATIONS TO YOU OF ANY KIND, WHETHER EXPRESS OR IMPLIED. WHERE LEGALLY LIABILITY CANNOT BE EXCLUDED FOR PRE-RELEASE AND/OR EVALUATION SOFTWARE, BUT IT MAY BE LIMITED, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE SUM OF FIFTY DOLLARS (USD $50) IN TOTAL. If the Evaluation Software has a time-out feature, then the software will cease operation after the conclusion of the designated evaluation period. Access to any files created with the Evaluation Software is entirely at your risk.

4. Authentic Desktop Community Edition ("ADCE") Software Terms and Conditions

(a) License Grant and Keycode. Upon your acceptance of this AEULA, Altova grants you a non-exclusive, non-transferable limited license, without the right to grant sublicenses, to install and use a copy of ADCE software on your compatible personal computer or workstation for the purpose of viewing, distributing, sharing, and editing of XML files solely in connection with STYLEVISION® Power Stylesheets as further provided herein. You will receive a key code that will enable you to activate or operate the ADCE software. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova. You may make one (1) backup and one (1) archival copy of the ADCE software, provided your backup and archival copies are not installed or used on any computer and further provided that all such copies shall bear the original and unmodified copyright, patent and other intellectual property markings that appear on or in the ADCE. You may install one (1) copy of such Setup Program for ADCE software on a computer file server within your internal network for the sole and exclusive purpose of installing the ADCE software to an unlimited number of client computers on your internal network. No other server or network use of the ADCE software is permitted, including but not limited to using the ADCE software (i) either directly or through commands, data or instructions

 © 2015 Altova GmbH

from or to another computer or (ii) for internal network, internet or web hosting services.

(b) Distribution. Upon your acceptance of this AEULA as part of your use of the ADCE software, and subject to your ongoing compliance with its terms and conditions, Altova hereby grants ADCE software users a non-exclusive, non-transferable, limited license, without the right to grant sublicenses, to reproduce the Setup Program for the ADCE software and distribute the Setup Program for the ADCE software in executable form to end users in the manner hereinafter provided. You may distribute the Setup Program for the ADCE software to any third party electronically or via download from the website or on physical media such as CD-ROMS or diskettes as part of or in conjunction with products that you have developed.

(c) ADCE Software Specific Restrictions. In addition to the restrictions and obligations provided in other sections of this AEULA, your license to distribute the Setup Program for the ACDE software is further subject to all of the following restrictions: (i) ACDE software shall only be licensed and not sold, (ii) you may not make the ACDE software available as a stand-alone product and if distributed as part of a product bundle you may charge for the product bundle provided that you license such product bundle at the same or lower fee at which you license any reasonably equivalent product bundle which does not include the ACDE software, (iii) you must use the Setup Program for ACDE provided by Altova AS IS and may not impair, alter or remove Altova's AEULA, (which will appear in the installation process and which an end user must accept in order to be able to install or operate the ACDE software) or any other files, and (iv) you may not combine the ACDE software with your product in such a way that your product modifies or generates Stylevision Power Stylesheets.

(d) Applicable AEULA Terms. The terms and conditions set forth in Sections 4, 6 and 7 apply to the ADCE software.

5. Authentic Browser-Plugin Community Edition ("ABCE") Software Terms and Conditions

(a) License Grant and Distribution. Upon your acceptance of this AEULA, Altova grants you a non-exclusive, non-transferable limited license, without the right to grant sublicenses, to use and develop web pages, web applications, or applications that include the ABCE software, to reproduce the ABCE software and to distribute the ABCE software in executable form in the manner hereinafter provided to end users for the purpose of viewing, sharing and editing XML files solely in connection with StyleVision® Power Stylesheets as further provided herein. You may install the ABCE software on a web server within your network for the purpose of downloading and installing the ABCE software (to an unlimited number of client computers on your internal network). You may distribute the ABCE software to any third party electronically or via download from the website or on physical media such as CD-ROMS or diskettes as part of or in conjunction with products that you have developed.

(b) ABCE Software Specific Restrictions. In addition to the restrictions and obligations provided in other sections of this AEULA, your license to distribute ABCE software is further subject to all of the following restrictions: (i) the ABCE software shall only be licensed and not sold; (ii) you may not make the ABCE software available as a stand-alone product and if distributed as part of a product bundle you may charge for the product bundle provided that you license such product bundle at the same or lower fee at which you license any reasonably equivalent product bundle which does not include the ABCE software; (iii) you must use the ABCE software provided by Altova AS IS and may not impair, alter or remove Altova's AEULA (which will appear in the installation process and which an end user must accept in order to be able to install or operate the ABCE software) or any other files; (iv) other Altova products cannot be distributed under this AEULA; and (v) you may not combine the ABCE software with your

product in such a way that your product modifies or generates Stylevision Power Stylesheet(s).

(c) Applicable AEULA Terms. The terms and conditions set forth in Sections 5, 6 and 7 apply to the ABCE software.

6. Authentic Community Editions (ADCE and ABCE) Software Additional Terms and Conditions

The terms set forth in Section 6 are applicable to the ADCE and ABCE software licenses and are in addition to the specific terms applicable to those software licenses.

(a) Use Limitation. The ADCE and ABCE software are licensed and distributed by Altova for viewing, distributing, sharing, and editing of XML files solely in connection with StyleVision® Power Stylesheets, defined as .sps files that are template files developed by Altova or its customers using Altova's StyleVision® product. You are not authorized to integrate or use the ADCE or ABCE software with (i) any StyleVision® Power Stylesheet(s) not developed in accordance with the Altova Software License Agreement available at http://www.altova.com/eula or (ii) other software or enhancement that uses Inter Application Communication (IAC) to programmatically interface with ADCE or ABCE software for the purpose of enabling additional functionality normally not available in ADCE or ABCE software or providing functionality that competes with other Altova products.

(b) Warranty Disclaimer. THE ADCE OR ABCE SOFTWARE IS PROVIDED TO YOU FREE OF CHARGE, AND ON AN "**AS-IS**" BASIS. ALTOVA PROVIDES NO WARRANTIES FOR THE ADCE OR ABCE SOFTWARE. TO THE MAXIMUM EXTENT PERMITTED BY LAW, ALTOVA AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES AND REPRESENTATIONS, WHETHER EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, SATISFACTORY QUALITY, INFORMATIONAL CONTENT, OR ACCURACY, QUIET ENJOYMENT, TITLE, AND NON- INFRINGEMENT. ALTOVA DOES NOT WARRANT THAT THE ADCE OR ABCE SOFTWARE IS ERROR-FREE OR WILL OPERATE WITHOUT INTERRUPTION. IF MANDATORILY APPLICABLE LAW REQUIRES ANY WARRANTIES WITH RESPECT TO THE ADCE OR ABCE SOFTWARE, ALL SUCH WARRANTIES ARE LIMITED IN DURATION TO THIRTY (30) DAYS FROM THE DATE OF INSTALLATION OR BEGIN OF USE, WHATEVER IS THE EARLIER. SOME STATES OR JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS AND YOU MAY ALSO HAVE OTHER LEGAL RIGHTS THAT VARY FROM STATE TO STATE OR FROM JURISDICTION TO JURISDICTION. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR THE ACCURACY AND ADEQUACY OF THE ADCE OR ABCE SOFTWARE FOR YOUR INTENDED USE AND YOU WILL INDEMNIFY AND HOLD HARMLESS ALTOVA FROM ANY 3RD PARTY SUIT TO THE EXTENT BASED UPON THE ACCURACY AND ADEQUACY OF THE ADCE OR ABCE SOFTWARE IN YOUR USE.

(c) Limitation of Liability. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL ALTOVA OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE AUTHENTIC SOFTWARE, THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, OR ANY PROVISION OF THIS AEULA, EVEN IF ALTOVA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. WHERE LEGALLY, LIABILITY CANNOT BE EXCLUDED, BUT MAY BE

 © 2015 Altova GmbH

LIMITED, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE SUM OF FIFTY DOLLARS (USD $50) IN TOTAL. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY, THE ABOVE LIMITATION MAY NOT APPLY TO YOU. IN SUCH STATES AND JURISDICTIONS, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE GREATEST EXTENT PERMITTED BY LAW. THE FOREGOING LIMITATIONS ON LIABILITY ARE INTENDED TO APPLY TO THE WARRANTIES AND DISCLAIMERS ABOVE AND ALL OTHER ASPECTS OF THIS AEULA.

(d) Support. Altova is not obliged to provide technical support with respect to the ADCE or ABCE software, that is provided on an AS IS basis. To the extent that Altova in its sole discretion does provide support for these products, the technical support will be provided in the manner detailed in Section 3(b) of this AEULA and subject to all requirements therein contained.

7. Authentic Software (ADEE, ABEE, ADCE and ABCE) Additional Terms and Conditions

The terms set forth in Section 7 are applicable to the ADEE, ABEE, ADCE and ABCE software licenses and are in addition to the specific terms applicable to those software licenses.

(a) Title. Title to the Authentic Software is not transferred to you. Ownership of all copies of the Authentic Software and of copies made by you is vested in Altova, subject to the rights of use or distribution, as applicable, granted to you in this AEULA. All rights not specifically granted in this AEULA are reserved by Altova.

(b) Acknowledgement of Altova's Rights. You acknowledge that the Authentic Software and any copies that you are authorized by Altova to make are the intellectual property of and are owned by Altova and its suppliers. The structure, organization and code of the Authentic Software are the valuable trade secrets and confidential information of Altova and its suppliers. The Authentic Software is protected by copyright, including without limitation by United States Copyright Law, international treaty provisions and applicable laws in the country in which it is being used. You acknowledge that Altova retains the ownership of all patents, copyrights, trade secrets, trademarks and other intellectual property rights pertaining to the Authentic Software, and that Altova's ownership rights extend to any images, photographs, animations, videos, audio, music, text and "applets" incorporated into the Authentic Software and all accompanying printed materials. You will take no actions which adversely affect Altova's intellectual property rights in the Authentic Software. Trademarks shall be used in accordance with accepted trademark practice, including identification of trademark owners' names. Trademarks may only be used to identify printed output produced by the Authentic Software, and such use of any trademark does not give you any right of ownership in that trademark. XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™, and Altova® are trademarks of Altova GmbH (registered in numerous countries). Unicode and the Unicode Logo are trademarks of Unicode, Inc. Windows, Windows XP, Windows Vista, and Windows 7 are trademarks of Microsoft. W3C, CSS, DOM, MathML, RDF, XHTML, XML and XSL are trademarks (registered in numerous countries) of the World Wide Web Consortium (W3C); marks of the W3C are registered and held by its host institutions, MIT, INRIA and Keio. Except as expressly stated above, this AEULA does not grant you any intellectual property rights in the Authentic Software. Notifications of claimed copyright infringement should be sent to Altova's copyright agent as further provided on the Altova Web Site.

(c) Common Restrictions.

(i) You may not reverse engineer, decompile, disassemble or otherwise attempt to

discover the source code, underlying ideas, underlying user interface techniques or algorithms of the Authentic Software by any means whatsoever, directly or indirectly, or disclose any of the foregoing, except to the extent you may be expressly permitted to decompile under applicable law in the European Union, if it is essential to do so in order to achieve operability of the Authentic Software with another software program, and you have first requested Altova to provide the information necessary to achieve such operability and Altova has not made such information available. Altova has the right to impose reasonable conditions and to request a reasonable fee before providing such information. Any information supplied by Altova or obtained by you, as permitted hereunder, may only be used by you for the purpose described herein and may not be disclosed to any third party or used to create any software which is substantially similar to the expression of the Authentic Software. Requests for information from users in the European Union with respect to the above should be directed to the Altova Customer Support Department. You may not loan, rent, lease, sublicense, distribute or otherwise transfer all or any portion of the Authentic Software to third parties except to the limited extent expressly provided in this AEULA.

(ii) You may not copy, distribute, or make derivative works of the Authentic Software except as expressly set forth above, and any copies that you are permitted to make pursuant to this Authentic EULA must contain the same copyright, patent and other intellectual property markings that appear on or in the Authentic Software. You may not modify, adapt or translate the Authentic Software. You may not, directly or indirectly, encumber or suffer to exist any lien or security interest on the Authentic Software; knowingly take any action that would cause the Authentic Software to be placed in the public domain; or use the Authentic Software in any computer environment not specified in this Authentic EULA. You will comply with applicable law and Altova's instructions regarding the use of the Authentic Software. You agree to notify your employees and agents who may have access to the Authentic Software of the restrictions contained in this AEULA and to ensure their compliance with these restrictions. You may not alter or modify the Authentic Software or create a new installer for the Authentic Software.

(d) Authentic Software Activation, Updates, Metering and Data Use.

(i) Altova has a built-in license metering module that helps you to avoid any unintentional violation of this AEULA. Altova may use your internal network for license metering between installed versions of the Authentic Software. **Altova's Authentic Software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of installation, registration, use, or update to an Altova-operated license server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the Authentic Software and to improve customer service. Activation is based on the exchange of license related data between your computer and the Altova license server. You agree that Altova may use these measures and you agree to follow any applicable requirements. You further agree that use of license key codes that are not or were not generated by Altova and lawfully obtained from Altova or an authorized reseller as part of an effort to activate or use the Authentic Software violates Altova's intellectual property rights as well as the terms of this AEULA. You agree that efforts to circumvent or disable Altova's copyright protection mechanisms or license management mechanism violate Altova's intellectual property rights as well as the terms of this AEULA. Altova expressly reserves the rights to seek all available legal and equitable remedies to prevent such actions and to recover lost profits, damages and costs.**

(ii) Altova provides a new LiveUpdate notification service to you, which is free of charge. Altova may use your internal network and Internet connection for the purpose of transmitting license-related data to an Altova-operated LiveUpdate server to validate your license at appropriate intervals and determine if there is any update available for you. The terms and

conditions of the Privacy Policy are set out in full at http://www.altova.com/privacy and are incorporated by reference into this AEULA. By your acceptance of the terms of this AEULA or use of the Authentic Software, you authorize the collection, use and disclosure of information collected by Altova for the purposes provided for in this AEULA and/or the Privacy Policy.. Altova has the right in its sole discretion to amend this provision of the AEULA and/or Privacy Policy at any time. You are encouraged to review the terms of the Privacy Policy as posted on the Altova Web site from time to time.

(iii) Notice to European Users. Please note that the information as described in paragraph 7(d) above may be transferred outside of the European Economic Area, for purposes of processing, analysis, and review, by Altova, Inc. a company located in Beverly, Massachusetts, U.S.A., or its subsidiaries or Altova's subsidiaries or divisions, or authorized partners, located worldwide. You are advised that the United States uses a sectoral model of privacy protection that relies on a mix of legislation, governmental regulation, and self-regulation. You are further advised that the Council of the European Union has found that this model does not provide "adequate" privacy protections as contemplated by Article 25 of the European Union's Data Directive. (Directive 95/46/EC, 1995 O.J. (L 281) 31). Article 26 of the European Union's Data Directive allows for transfer of personal data from the European Union to a third country if the individual has unambiguously given his consent to the transfer of personal information, regardless of the third country's level of protection. By agreeing to this AEULA, you consent to the transfer of all such information to the United States and the processing of that information as described in this AEULA and the Privacy Policy.

(e) Disclaimer. THE AUTHENTIC SOFTWARE IS NEITHER GUARANTEED NOR WARRANTED TO BE ERROR-FREE NOR SHALL ANY LIABILITY BE ASSUMED BY ALTOVA IN THIS RESPECT. NOTWITHSTANDING ANY SUPPORT FOR ANY TECHNICAL STANDARD, THE AUTHENTIC SOFTWARE IS NOT INTENDED FOR USE IN OR IN CONNECTION WITH, WITHOUT LIMITATION, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION, COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL EQUIPMENT, MEDICAL DEVICES OR LIFE SUPPORT SYSTEMS, MEDICAL OR HEALTH CARE APPLICATIONS, OR OTHER APPLICATIONS WHERE THE FAILURE OF THE AUTHENTIC SOFTWARE OR ERRORS IN DATA PROCESSING COULD LEAD TO DEATH, PERSONAL INJURY OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR THE ACCURACY AND ADEQUACY OF THE AUTHENTIC SOFTWARE AND ANY DATA GENERATED OR PROCESSED BY THE SOFTWARE FOR YOUR INTENDED USE AND YOU WILL DEFEND, INDEMNIFY AND HOLD ALTOVA, ITS OFFICERS AND EMPLOYEES HARMLESS FROM ANY 3RD PARTY CLAIMS, DEMANDS, OR SUITS THAT ARE BASED UPON THE ACCURACY AND ADEQUACY OF THE AUTHENTIC SOFTWARE IN YOUR USE OR ANY DATA GENERATED BY THE AUTHENTIC SOFTWARE IN YOUR USE.

(f) Restricted Rights Notice and Export Restrictions. The Authentic Software was developed entirely at private expense and is commercial computer software provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the U.S. Government or a U.S. Government contractor or subcontractor is subject to the restrictions set forth in this Agreement and as provided in FAR 12.211 and 12.212 (48 C.F.R. §12.211 and 12.212) or DFARS 227. 7202 (48 C.F.R. §227-7202) as applicable. Consistent with the above, as applicable, Commercial Computer Software and Commercial Computer Documentation licensed to U.S. government end users only as commercial items and only with those rights as are granted to all other end users under the terms and conditions set forth in this AEULA. Manufacturer is Altova GmbH, Rudolfsplatz, 13a/9, A-1010 Vienna, Austria/EU. You may not use or otherwise export or re-export the Authentic Software or documentation except as authorized by United States law and the laws of the jurisdiction in which the Authentic Software was obtained. In particular, but without limitation, the Authentic Software or Documentation may not be exported or re-exported (i) into (or

to a national or resident of) any U.S. embargoed country or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals or the U.S. Department of Commerce's Table of Denial Orders. By using the Software, you represent and warrant that you are not located in, under control of, or a national or resident of any such country or on any such list.

(g) Termination. Without prejudice to any other rights or remedies of Altova, this AEULA may be terminated (i) by you giving Altova written notice of termination or (ii) by Altova, at its option, giving you written notice of termination or (iii) Altova giving you written notice of termination if you fail to comply with the terms and conditions of the AEULA. This AEULA automatically terminates upon the expiration of the ABEE License Term. Upon any termination or expiration of this AEULA, you must cease all use of Authentic Software, licensed hereunder, destroy all copies then in your possession or control and take such other actions as Altova may reasonably request to ensure that no copies of the Authentic Software remain in your possession or control. The terms and conditions set forth in Sections 1(e), 2(c)-(d), 3(c)-(d), 4(c)-(d), 5(b)-(c), 6(b)-(c) and 7 survive termination of this AEULA as applicable.

(h) Third Party Software. The Authentic Software may contain third party software which requires notices and/or additional terms and conditions. Such required third party software notices and/or additional terms and conditions are located at our website at http://www.altova.com/legal_3rdparty.html and are made a part of and incorporated by reference into this AEULA. By accepting this AEULA, you are also accepting the additional terms and conditions, if any, set forth therein.

(i) General Legal Provisions. This AEULA contains the entire agreement and understanding of the parties with respect to the subject matter hereof, and supersedes all prior written and oral understandings of the parties with respect to the subject matter hereof. Any notice or other communication given under this AEULA shall be in writing and shall have been properly given by either of us to the other if sent by certified or registered mail, return receipt requested, or by overnight courier to the address shown on Altova's Web site for Altova and the address shown in Altova's records for you, or such other address as the parties may designate by notice given in the manner set forth above. This AEULA will bind and inure to the benefit of the parties and our respective heirs, personal and legal representatives, affiliates, successors and permitted assigns. The failure of either of us at any time to require performance of any provision hereof shall in no manner affect such party's right at a later time to enforce the same or any other term of this AEULA. This AEULA may be amended only by a document in writing signed by both of us. In the event of a breach or threatened breach of this AEULA by either party, the other shall have all applicable equitable as well as legal remedies. Each party is duly authorized and empowered to enter into and perform this AEULA. If, for any reason, any provision of this AEULA is held invalid or otherwise unenforceable, such invalidity or unenforceability shall not affect the remainder of this AEULA, and this AEULA shall continue in full force and effect to the fullest extent allowed by law. The parties knowingly and expressly consent to the foregoing terms and conditions.

(i) If you are located in the European Union and are using the Authentic Software in the European Union and not in the United States, then this AEULA will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Authentic Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht, Wien (Commercial Court, Vienna) in connection with any such dispute or claim.

© 2015 Altova GmbH

(ii) If you are located in the United States or are using the Authentic Software in the United States then this AEULA will be governed by and construed in accordance with the laws of the Commonwealth of Massachusetts, USA (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Authentic Software resides in the federal or state courts of the Commonwealth of Massachusetts and you further agree and expressly consent to the exercise of personal jurisdiction in the federal or state courts of the Commonwealth of Massachusetts in connection with any such dispute or claim.

(iii) If you are located outside of the European Union or the United States and are not using the Authentic Software in the United States, then this AEULA will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Authentic Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht Wien (Commercial Court, Vienna) in connection with any such dispute or claim. This AEULA will not be governed by the conflict of law rules of any jurisdiction or the United Nations Convention on Contracts for the International Sale of Goods, the application of which is expressly excluded.

Last Updated: 2011-10-01

Índice

.

.NET,
diferencias con la versión independiente de Authentic Desktop, 129
integración de Authentic Desktop con, 127

A

Abrir,
archivo, 144
Acceso rápido, 258
asignar/eliminar, 258
Activar el software, 295
Activar/desactivar, 285, 286, 287, 290
Administrador del control de código fuente, 191
Alias,
ver Recursos globales, 86
Analizador,
integrado en los productos de Altova, 727
XSLT, 277
Analizador XML,
información sobre, 727
Analizador XML de Altova,
información sobre, 727
Ancho de celda máximo, 276
Ancho de tabla óptimo, 276
Anexar,
fila (en la vista Authentic), 234
Anexos, 724
API,
accessing, 701
documentation, 373
overview, 375
API de XMLSpy,
documentation, 373
overview, 375
Aplicar, 273
Application,
ActiveDocument, 413
AddMacroMenuItem, 413
AddXSLT_XQParameter, 414
Application, 414
ClearMacroMenu, 414
CurrentProject, 415
Dialogs, 415
Documents, 416
GetDatabaseImportElementList, 416
GetDatabaseSettings, 417
GetDatabaseTables, 417
GetExportSettings, 418
GetTextImportElementList, 418
GetTextImportExportSettings, 419
GetXSLT_XQParameterCount, 420
GetXSLT_XQParameterName, 420
GetXSLT_XQParameterXPath, 420
ImportFromDatabase, 420
ImportFromSchema, 421
ImportFromText, 422
ImportFromWord, 423
NewProject, 424
OnBeforeOpenDocument, 411
OnBeforeOpenProject, 412
OnDocumentOpened, 412
OnProjectOpened, 413
OpenProject, 424
Parent, 425
Quit, 425
ReloadSettings, 425
RemoveXSLT_XQParameter, 426
RunMacro, 426
ScriptingEnvironment, 426
ShowApplication, 427
ShowForm, 427
URLDelete, 428
URLMakeDirectory, 428
WarningNumber, 429
WarningText, 429
Application-level,
integration of Authentic Desktop, 654
Archivo, 273
abrir, 144
cerrar, 152
codificación predeterminada, 277
codificar, 151
crear nuevo, 142
guardar, 153
guardar por correo electrónico, 159
opciones de impresión, 161

© 2015 Altova GmbH

Archivo, 273
opciones para abrir, 273
pestaña, 273
Archivo nuevo,
crear, 142
Archivo PXF,
generar documentos de salida desde la vista Authentic, 236
Archivo XML de plantilla,
en la vista Authentic, 24
Archivo XML de recursos globales, 87
Archivos,
agregarlos al control de código fuente, 184
usados recientemente, 163
Archivos ASPRJ, 305
Archivos de plantilla,
para documentos nuevos, 142
Archivos no XML, 275
Archivos usados recientemente,
lista de, 163
Asignar,
tecla de acceso rápido a un comando, 258
ATL,
archivos de muestra para complementos, 358
Atributos (ayudante de entrada),
en la vista Authentic, 47
Authentic Desktop,
integration, 653
manual del usuario, 2
Authentic Desktop API,
accessing, 701
Authentic Desktop command table, 681
Authentic Desktop integration,
example of, 662
AuthenticDesktopCommand,
in AuthenticDesktopControl, 703
AuthenticDesktopCommands,
in AuthenticDesktopControl, 705
AuthenticDataTransfer,
dropEffect, 432
getData, 432
ownDrag, 433
type, 433
AuthenticDesktopControl, 706
documentation of, 653
example of integration at application level, 662
examples of integration at document level, 660
integration at application level, 654
integration at document level, 655, 656, 657, 658, 659
integration using C#, 661
integration using HTML, 663
object reference, 702
AuthenticDesktopControlDocument, 713
AuthenticDesktopControlPlaceHolder, 719
AuthenticRange,
AppendRow, 438
Application, 439
CanPerformAction, 439
CanPerformActionWith, 439
Close, 440
CollapsToBegin, 440
CollapsToEnd, 440
Copy, 441
Cut, 441
Delete, 441
DeleteRow, 442
DuplicateRow, 442
ExpandTo, 443
FirstTextPosition, 443
FirstXMLData, 444
FirstXMLDataOffset, 445
GetElementAttributeNames, 446
GetElementAttributeValue, 447
GetElementHierarchy, 447
GetEntityNames, 447
Goto, 448
GotoNext, 448
GotoNextCursorPosition, 449
GotoPrevious, 450
GotoPreviousCursorPosition, 450
HasElementAttribute, 451
InsertEntity, 451
InsertRow, 452
IsCopyEnabled, 452
IsCutEnabled, 452
IsDeleteEnabled, 453
IsEmpty, 453
IsEqual, 453
IsFirstRow, 453
IsInDynamicTable, 454
IsLastRow, 454
IsPasteEnabled, 454
IsTextStateApplied, 455
LastTextPosition, 455
LastXMLData, 456
LastXMLDataOffset, 457
MoveBegin, 458

AuthenticRange,
MoveEnd, 458
MoveRowDown, 459
MoveRowUp, 459
Parent, 459
Paste, 460
PerformAction, 460
Select, 461
SelectNext, 461
SelectPrevious, 462
SetElementAttributeValue, 463
SetFromRange, 464
Text, 464
AuthenticView, 481
Application, 474
AsXMLString, 474
DocumentBegin, 476
DocumentEnd, 476
Event, 477
Goto, 478
IsRedoEnabled, 478
IsUndoEnabled, 479
MarkupVisibility, 479
OnBeforeCopy, 466
OnBeforeCut, 467
OnBeforeDelete, 467
OnBeforeDrop, 468
OnBeforePaste, 468
OnDragOver, 469
OnKeyboardEvent, 470
OnMouseEvent, 471
OnSelectionChanged, 471
Parent, 479
Print, 479
Redo, 480
Selection, 480
Undo, 481
WholeDocument, 482
XMLDataRoot, 482
Ayuda,
contenido, 293
mapa de teclado, 294
Ayudante de entrada, 290
Ayudantes de entrada, 14, 287
activar/desactivar, 289

B

Barra de estado, 16
Barra de herramientas, 16, 257
activar/desactivar, 257
agregar comando a, 255
crear nueva, 257
mostrar iconos grandes, 269
restablecer barra de herramientas y comandos de menú, 257
Barra de menú, 16
Bases de datos,
edición en la vista Authentic, 228
ver también BD, 69
BD, 69, 71
consultas en la vista Authentic, 69
crear consultas, 71
editar en la vista Authentic, 69, 76
filtrar opciones de visualización en la vista Authentic, 71
navegar por tablas en la vista Authentic, 70
parámetros en consultas de BD, 71
BD XML,
cargar filas de datos XML nuevas, 70
cargar nueva fila de datos en la vista Authentic, 230
Big-endian, 277
Buscar, 169
texto en el documento, 168
y reemplazar texto en el documento, 169
Buscar cambios, 273

C

C#,
integration of Authentic Desktop, 661
Cambiar de vista,
a la vista Authentic, 41
Capítulos, 293
Catálogo,
Oasis XML, 215
Centro de soporte técnico, 297
Class ID,
in Authentic Desktop integration, 662
Clave de evaluación,
para su software de Altova, 295

© 2015 Altova GmbH

CodeGeneratorDlg,
Application, 483
CPPSettings_DOMType, 484
CPPSettings_LibraryType, 485
CPPSettings_UseMFC, 485
CSharpSettings_ProjectType, 485
OutputPath, 486
OutputPathDialogAction, 486
OutputResultDialogAction, 486
Parent, 487
ProgrammingLanguage, 487
PropertySheetDialogAction, 487
TemplateFileName, 488
Codificación,
de archivos, 151
predeterminada, 277
Códigos clave,
para su software de Altova, 295
Color de sintaxis, 275, 276
Comando, 265
agregar a menú/barra de herramientas, 255
eliminar de menú, 265
menú contextual, 265
restablecer menú, 265
Comandos,
listado del mapa de teclado, 294
Comandos de menú, 140
COM-API,
documentation, 373
Compatibilidad con Unicode,
de los productos de Altova, 729
Complemento,
archivos de muestra ATL, 358
configuración de la interfaz del usuario, 354
registro, 352
XMLSPY, 351
Complemento de Authentic para Eclipse,
instalar, 131
Complemento de Authentic para VS .NET,
instalar, 128
Comprar software de Altova, 295
Comprobación de formato XML, 213
Configuración, 19, 273
Configuración activa,
para recursos globales, 254
Configuración del programa, 273
Configuraciones,
de un recurso global, 87
Configuraciones en recursos globales, 104
Configurar,
interfaz de usuario de XMLSpy, 354
Configurar impresión, 162
Conformidad con XML, 275
Consultas,
para visualizar BD en la vista Authentic, 71
Contenido, 293
Contrato de licencia para el usuario final, 731
Control de código fuente, 105, 280
abrir proyecto, 178
actualizar estado, 191
agregar al control de código fuente, 184
anular desprotección, 184
cambiar de proveedor, 191
compartir desde, 186
desproteger, 182
habilitar, deshabilitar, 179
mostrar diferencias, 189
mostrar historial, 188
obtener archivos, 180
obtener la versión más reciente, 180
propiedades, 190
proveedores compatibles, 178
quitar del, 185
Controladores de eventos,
en el proyecto de scripting, 326
información general, 312
Controles ActiveX,
compatibilidad, 353
Copiar (comando), 166
Correo electrónico,
enviar archivos por, 159
Cortar (comando), 166
CPU,
carga - acelerar actualización de estado en segundo plano, 105
CR & LF, 273
CustomCatalog, 215
CVS, 105

D

DatabaseConnection,
ADOConnection, 489
AsAttributes, 490

DatabaseConnection,
CreateMissingTables, 490
CreateNew, 491
DatabaseKind, 491
ExcludeKeys, 491
File, 492
IncludeEmptyElements, 492
NumberDateTimeFormat, 493
ODBCConnection, 493
SQLSelect, 494
TextFieldLen, 494
Declaraciones globales, 316
información general, 312
Delete,
Application.URLDelete, 428
Depurar macros, 336
Descripción de la IGU, 7
Descripción de la interfaz gráfica, 7
Deshabilitar validación automática, 275
Deshacer (comando), 165
Dialogs,
Application, 496
CodeGeneratorDlg, 496
DTDSchemaGeneratorDlg, 498
FileSelectionDlg, 497
GenerateSampleXMLDlg, 497
Parent, 497
SchemaDocumentationDlg, 497
Diccionario, 245
añadir diccionario personalizado, 245
modificar, 245
revisión ortográfica, 245
Diccionario personalizado, 245
directories,
creating with Application.URLMakeDirectory, 428
Distribución,
de productos de software de Altova, 731, 732, 733
Document, 511
Application, 503
AssignDTD, 504
AssignSchema, 504
AssignXSL, 504
AssignXSLFO, 505
AuthenticView, 505
Close, 506
ConvertDTDOrSchema, 506
CreateChild, 508
CreateSchemaDiagram, 509
CurrentViewMode, 509
DataRoot, 509
DocEditView, 510
Encoding, 510
EndChanges, 510
ExecuteXQuery, 511
ExportToDatabase, 511
ExportToText, 512
FullName, 514
GenerateDTDOrSchema, 514, 515
GenerateProgramCode, 515
GenerateSampleXML, 515
GenerateSchemaDocumentation, 515
GetExportElementList, 516
GetPathName, 517
GridView, 517
IsModified, 517
IsValid, 518
IsWellFormed, 519
Name, 519
OnBeforeCloseDocument, 501
OnBeforeSaveDocument, 500
OnBeforeValidate, 502
OnCloseDocument, 502
OnViewActivation, 503
Path, 520
RootElement, 520
Save, 520
SaveAs, 521
Saved, 521
SaveInString, 521
SaveToURL, 522
SetActiveDocument, 522
SetEncoding, 522
SetExternalIsValid, 524
SetPathName, 524
StartChanges, 524
SwitchViewMode, 525
Title, 525
TransformXSL, 526
TransformXSLFO, 526
UpdateViews, 527
UpdateXMLData, 527
XQuery, 511
Document-level,
examples of integration of XMLSpy, 660
integration of Authentic Desktop, 656, 657, 658, 659
integration of AuthenticDesktopControl, 655

© 2015 Altova GmbH

Documento,
revisión ortográfica, 245
Documento de salida HTML,
generar en la vista Authentic a partir del archivo PXF, 236
Documento de salida PDF,
generar en la vista Authentic a partir del archivo PXF, 236
Documento de salida RTF,
generar en la vista Authentic a partir del archivo PXF, 236
Documento de salida Word 2007+,
generar en la vista Authentic a partir del archivo PXF, 236
Documento XML,
abrir en la vista Authentic, 24
Documentos en la ventana principal, 9
Documents,
Count, 528
Item, 529
NewAuthenticFile, 529
NewFile, 529
NewFileFromText, 530
OpenAuthenticFile, 530
OpenFile, 530
OpenURL, 531
OpenURLDialog, 532
DTD, 273, 275
DTDSchemaGeneratorDlg,
Application, 533
AttributeTypeDefinition, 533
DTDSchemaFormat, 534
FrequentElements, 534
GlobalAttributes, 534
MaxEnumLength, 534
MergeAllEqualNamed, 535
OnlyStringEnums, 535
OutputPath, 535
OutputPathDialogAction, 535
Parent, 536
ResolveEntities, 536
TypeDetection, 536
ValueList, 536
Duplicar,
fila (en la vista Authentic), 234

E

Editado con XMLSPY, 273
Editar,
botón de macro, 269
Editor de scripting,
descripción de la interfaz gráfica del usuario, 307
ventana principal, 307
Editor predeterminado, 275
ElementList,
Count, 537
Item, 537
RemoveElement, 537
ElementListItem,
ElementKind, 539
FieldCount, 539
Name, 539
RecordCount, 539
Elementos (ayudante de entrada),
en la vista Authentic, 47
Elementos vacíos, 275
Eliminar, 255
acceso rápido, 258
barra de herramientas, 257
comando de la barra de herramientas, 255
comando del menú contextual, 265
fila (en la vista Authentic), 234
icono de la barra de herramientas, 255
Eliminar (comando), 166
En cascada,
ventanas, 282
En paralelo, 276
Entidades,
definir en la vista Authentic, 56, 81
insertar en la vista Authentic, 36, 56
Entidades (ayudante de entrada),
en la vista Authentic, 47
Entidades externas analizadas, 275
Entorno de scripting, 302
resumen, 304
Enumerations,
in AuthenticDesktopControl, 722
SPYAttributeTypeDefinition, 610
SPYAuthenticActions, 611
SPYAuthenticDocumentPosition, 612
SpyAuthenticElementActions, 613
SPYAuthenticElementKind, 614
SPYAuthenticMarkupVisibility, 615
SPYDatabaseKind, 617
SPYDialogAction, 618
SPYDOMType, 619
SPYDTDSchemaFormat, 620

© 2015 Altova GmbH

Enumerations,
SPYEncodingByteOrder, 621
SPYExportNamespace, 622
SPYFrequentElements, 624
SPYKeyEvent, 627
SPYLibType, 629
SPYLoading, 630
SPYMouseEvent, 631
SPYNumberDateTimeFormat, 632
SPYProgrammingLanguage, 633
SPYProjectItemTypes, 634
SPYProjectType, 635
SPYSampleXMLGenerationOptimization, 637
SPYSampleXMLGenerationSchemaOrDTDAssignment, 639
SPYSchemaDefKind, 640
SPYSchemaDocumentationFormat, 641
SPYTextDelimiters, 644
SPYTextEnclosing, 645
SPYTypeDetection, 646
SPYURLTapes, 647
SPYViewModes, 650
SPYVirtualKeyMask, 651
SPYXMLDataKind, 652
Esquema,
configuración, 273
Event, 411, 412, 413, 466, 467, 468, 469, 470, 471, 500, 501, 502, 503, 562, 563, 564
Eventos, 326
y controladores de eventos, 314
Eventos de la aplicación, 326
Eventos de la vista Authentic, 326
Eventos de la vista Cuadrícula, 326
Eventos de la vista Texto, 326
Eventos del documento, 326
Events, 384
Explorador, 275, 276
Vista, 85
Explorador (menú), 237
ExportSettings,
CreateKeys, 541
ElementList, 541
EntitiesToText, 541
ExportAllElements, 542
FromAttributes, 542
FromSingleSubElements, 542
FromTextValues, 542
IndependentPrimaryKey, 543
Namespace, 543
SubLevelLimit, 543
Expresiones regulares,
en la cadena de búsqueda, 168
Extensiones de archivo,
personalizar, 215

F

Favoritos, 293
Fechas,
cambiar manualmente, 80
Fila,
anexar (en la vista Authentic), 234
duplicar (en la vista Authentic), 234
eliminar (en la vista Authentic), 234
insertar (en la vista Authentic), 234
subir/bajar, 235
FileSelectionDlg,
Application, 544
DialogAction, 544
FullName, 545
Parent, 545
Formato de salida, 273
Formatos de imagen,
en la vista Authentic, 83
Formatos gráficos,
en la vista Authentic, 83
Formularios,
crear nuevos, 319
del proyecto de scripting, 318
establecer la secuencia de tabulación de los objetos, 321
información general, 312
invocar, 314
nombrar, 319
propiedades, 319
y comandos integrados, 339
y control de eventos, 324
y objetos de formulario, 321

G

Generate Sample XML, 610, 637, 639
GenerateSampleXMLDlg,

© 2015 Altova GmbH

GenerateSampleXMLDlg,
Application, 557
FillWithSampleData, 559
NonMandatoryAttributes, 559
NonMandatoryElements, 559
Parent, 560
RepeatCount, 560
TakeFirstChoice, 561
Global,
configuración, 273
Gramática, 275
GridView,
CurrentFocus, 564
Deselect, 565
IsVisible, 565
OnBeforeDrag, 562
OnBeforeDrop, 562
OnBeforeStartEditing, 563
OnEditingFinished, 563
OnFocusChanged, 564
Select, 565
SetFocus, 565
Grupo de inicio,
agregar (menú contextual), 269
Guardar archivos,
codificación de, 151

H

Hipervínculos,
navegar por la vista Authentic, 56
HTML,
integration of Authentic Desktop, 663
HTML example,
of AuthenticDesktopControl integration, 662

I

Icono,
agregar a menú/barra de herramientas, 255
mostrar en tamaño grande, 269
Imprimir,
desde la vista Authentic, 37
Información (ventana), 290
Información general, 725
Información legal, 731
Información rápida,
mostrar, 269
mostrar teclas de acceso rápido en, 269
Información sobre derechos de autor, 731
Información técnica, 725
Insertar,
fila (en la vista Authentic), 234
Integrating,
Authentic Desktop in applications, 653
Interfaz del usuario,
configurar usando complementos, 354
Internet, 297

J

Java, 665
JRE,
para el complemento de Authentic para Eclipse, 131
Juego de caracteres,
codificación, 277

L

Lenguaje,
cambiar el lenguaje de scripting, 314
Lenguaje de scripting, 314
Licencia,
información sobre, 731
para su software de Altova, 295
Línea de comandos, 298
Little-endian, 277
loading, 531

M

Macro,
botón, 269
Macros,
crear con el editor de scripts, 331
definir como macros automáticas en el editor de scripts, 331
depurar, 336

Macros,
editar con el editor de scripts, 331
ejecutar, 314, 333
funciones para, en declaraciones globales, 316
información general, 312
instrucciones de uso en el proyecto de scripting, 330
MainCatalog, 215
Manual del usuario, 2
Authentic Desktop, 4
Mapa de teclado, 294
Marcado,
en la vista Authentic, 41, 44
Marcado (en la vista Authentic),
mostrar en tamaño pequeño/grande/mixto, 233
ocultar, 233
Marcado en tamaño grande (en la vista Authentic), 233
Marcado en tamaño pequeño (en la vista Authentic), 233
Marcado mixto (en la vista Authentic), 233
Marcador, 293
Mensajes de validación, 15
Menú, 265
agregar/eliminar comando, 255
Authentic, 226
Ayuda, 292
Edición, 164
eliminar comandos de, 265
Herramientas, 244
personalizar, 265
predeterminado/XMLSPY, 265
Proyecto, 170
Ventanas, 281
XML, 212
XSL/XQuery, 218
Menú Archivo, 141
Menú Authentic, 226
edición de tablas dinámicas, 41
presentación del marcado, 41
Menú Ayuda, 292
Menú contextual,
comandos, 265
para personalización, 269
Menú Edición, 164
Menú Explorador, 237
Menú Herramientas, 244
Menú Proyecto, 170
Menú Ventanas, 281
Menú XML, 212
Menú XSL/XQuery, 218
Menús contextuales,
en la vista Authentic, 52
Microsoft® SharePoint® Server, 203
MIME, 275
Mosaico,
horizontal, 283
vertical, 284
Mostrar, 285, 286, 287, 290
Mostrar marcado en tamaño grande, 41, 44
Mostrar marcado en tamaño pequeño, 41, 44
Mostrar marcado mixto, 41, 44
Motores,
de los productos de Altova, 728
MS Visual Source Safe, 105
MSXML, 277
Multiusuario, 273

N

Notas sobre programación,
del proyecto de scripting, 337

O

OASIS,
Catálogo XML, 215
Ocultar, 285, 286, 287, 290
Ocultar marcado, 41, 44
Ocultar marcado (en la vista Authentic), 233
Ocultar ventanas automáticamente, 7
Opción Macro automática, 331
Opciones de impresión, 161
Opciones de ortografía, 249
Opciones de soporte, 21
Opciones para abrir,
archivos, 273
Overview,
of API de XMLSpy, 375

P

Paleta de objetos de formularios, 307

© 2015 Altova GmbH

Panel Árbol del proyecto de scripting, 307
Panel Propiedades y eventos, 307
Paquete de integración de Authentic, 128, 131
Parámetros,
en las consultas de BD, 71
pasarlos a la hoja de estilos a través de la interfaz, 222
Parámetros XSLT,
pasarlos a la hoja de estilos a través de la interfaz, 222
Parent, 519
Pegar,
como texto, 56
como XML, 56
Pegar (comando), 166
Pegar como,
Texto, 52
XML, 52
Período de evaluación,
de los productos de software de Altova, 731, 732, 733
Personalización, 19
Personalizar, 265
comandos de menú/barra de herramientas, 255
menú, 265
menú contextual, 265
menú contextual Personalizar, 269
Perspectiva Authentic en Eclipse, 137
Plantillas,
de documentos XML en la vista Authentic, 227
Plantillas en la vista Authentic, 24
Plataforma Eclipse,
perspectiva Authentic, 137
y Authentic Desktop, 130
y el paquete de integración de Authentic, 131
Plataformas,
para productos de Altova, 726
Predeterminada,
codificación, 277
Predeterminado,
menú, 265
Presentación, 276
Procesador XQuery,
de los productos de Altova, 728
Procesador XSL externo, 277
Procesadores XSLT,
de los productos de Altova, 728
Productos de Altova, 21
Propiedades de los objetos de formulario, 321
Proveedor,
de control de código fuente, 105
Proyecto,
abrir, 174
agregar archivo activo al, 196, 197
agregar archivos al, 193
agregar carpetas al, 198
agregar carpetas externas al, 199
agregar carpetas web externas al, 203
agregar recursos globales al, 194
agregar URL al, 195
agregarlo al control de código fuente, 184
cerrar, 176
crear nuevo, 173
guardar, 177
propiedades, 208
volver a cargar, 175
Proyecto (ventana), 290
Proyecto de scripting,
controladores de eventos, 312
controladores de eventos de la aplicación, 326
declaraciones globales, 312
declaraciones globales del, 316
formularios, 312
formularios del, 318
instrucciones para crear, 314
macros, 312
macros del, 330
notas sobre programación, 337
y eventos, 326
Proyecto de scripting global,
de Authentic Desktop, 305
Proyectos,
agregar archivos relacionados al, 197
introducción, 170
usados recientemente, 211
Proyectos de scripting,
para Authentic Desktop, 305
para proyecto de Authentic Desktop, 305
Proyectos de scripting de Altova, 305
PUBLIC,
identificador - catálogo, 215
PVCS Version Manager, 105

R

Recargar, 273
Recursos,

© 2015 Altova GmbH

Recursos,
acelerar actualización de estado en segundo plano, 105
Recursos globales, 86
activación de barras de herramientas, 257
cambiar de configuración, 104
configuración activa para, 254
definir, 87, 253
definir el tipo de archivo, 90
definir el tipo de base de datos, 98
definir el tipo de carpeta, 96
usar, 100, 104
usar tipo de archivo y tipo de carpeta, 101
Recursos globales de Altova,
ver Recursos globales, 86
Reemplazar,
texto, 168
texto en el documento, 169
Referencia para programadores, 300
Registrar,
complemento, 352
Registrar su software de Altova, 295
Registro,
configuración, 273
Rehacer (comando), 165
Repositorios, 105
Requisitos de memoria, 726
Restablecer,
acceso rápido, 258
barras de herramientas y comandos de menú, 257
comandos de menú, 265
Retorno de carro,
ver Tecla Entrar, 84
Revisar,
ortografía, 245
Revisión ortográfica, 245
diccionario personalizado, 245

S

Saltos de línea, 273
save, 522
schema, 421
SchemaDocumentationDlg,
AllDetails, 567
Application, 567
IncludeAll, 569
IncludeAttributeGroups, 569
IncludeComplexTypes, 569
IncludeGlobalElements, 570
IncludeGroups, 570
IncludeIndex, 571
IncludeLocalElements, 571
IncludeRedefines, 571
IncludeSimpleTypes, 572
OptionsDialogAction, 573
OutputFile, 573
OutputFileDialogAction, 573
OutputFormat, 574
Parent, 574
ShowAnnotations, 574
ShowAttributes, 574
ShowChildren, 575
ShowConstraints, 576
ShowDiagram, 575
ShowEnumerations, 575
ShowNamespace, 576
ShowPatterns, 576
ShowProgressBar, 577
ShowProperties, 577
ShowResult, 577
ShowSingleFacets, 578
ShowSourceCode, 578
ShowType, 578
ShowUsedBy, 578
Secciones CDATA,
insertar en la vista Authentic, 56
Segundo plano,
actualización de estado - aumentar el intervalo, 105
Seleccionar todo (comando), 167
Selector de fecha,
usar en la vista Authentic, 79
Servidor web, 297
SharePoint® Server, 203
Sistema de ayuda, 293
SO,
para productos de Altova, 726
Soporte técnico, 297
Soporte técnico de Altova, 21
SPS,
asignar a un archivo XML nuevo, 142
SpyProject,
CloseProject, 580
ProjectFile, 580
RootItems, 580

© 2015 Altova GmbH

SpyProject,
SaveProject, 581
SaveProjectAs, 581
SpyProjectItem,
ChildItems, 582
FileExtensions, 582
ItemType, 582
Name, 583
Open, 583
ParentItem, 583
Path, 583
ValidateWith, 583
XMLForXSLTransformation, 584
XSLForXMLTransformation, 584
XSLTransformationFileExtension, 584
XSLTransformationFolder, 584
SpyProjectItems,
AddFile, 585
AddFolder, 585
AddURL, 585
Count, 586
Item, 586
RemoveItem, 586
StarTeam, 105
Subir/bajar,
fila (vista Authentic), 235

T

Tabla,
generar automáticamente, 275
Tablas,
editar tablas dinámicas (SPS), 41
en la vista Authentic, 29
Tablas (SPS) dinámicas en la vista Authentic,
uso de, 62
Tablas (SPS) estáticas en la vista Authentic,
uso de, 62
Tablas dinámicas,
edición, 41
Tablas en la vista Authentic,
iconos para editar tablas XML, 67
usar tablas SPS (estáticas y dinámicas), 62
usar tablas XML, 63
uso de, 61
Tablas SPS,
editar tablas dinámicas, 41
Tablas SPS en la vista Authentic,
uso de, 62
Tablas XML en la vista Authentic,
iconos para editar, 67
uso de, 63
Tamaño de la fuente,
en la vista Explorador, 242
Tecla de acceso rápido, 258
Tecla Entrar,
efectos de usar, 84
Teclas de acceso rápido, 258
mostrar en información rápida, 269
Tema,
ver en la tabla de contenido, 293
terminate, 425
TextImportExportSettings,
DestinationFolder, 587
EnclosingCharacter, 587
Encoding, 588
EncodingByteOrder, 588
FieldDelimiter, 588
FileExtension, 588
HeaderRow, 588
ImportFile, 589
Texto,
buscar en el documento, 168
buscar y reemplazar, 169
editar en la vista Authentic, 56
formato en la vista Authentic, 56
Tipos de archivo, 275
Título de la pantalla, 276
Transformación,
ver Transformación XSLT, 220
Transformación XSLT, 219, 220
en FO, 220
en PDF, 220

U

Ubicaciones del archivo SPP, 170
UCS-2, 277
URL, 428, 522, 531, 532
enviar por correo electrónico, 159
Uso de Internet,
en los productos de Altova, 730

UTF-16, 277

V

Validación, 19, 215
Validación automática, 275
Validador,
de los productos de Altova, 727
Valores de atributo,
introducir en la vista Authentic, 35
Velocidad,
acelerar actualización de estado en segundo plano, 105
Ventana, 290
Ayudantes de entrada, 287, 290
Información, 286, 290
Proyecto, 285, 290
Ventana acoplable, 285, 286
Ventana Información, 13, 286
Ventana Mensajes, 15
Ventana principal, 9
Ventana Proyecto, 11, 285
activar/desactivar, 289
Ventanas,
abrir, 291
administrar la visualización de, 7
en cascada, 282
en mosaico horizontal, 283
en mosaico vertical, 284
flotantes, acopladas, en pestañas, 7
ocultar automáticamente, 7
Ventanas acopladas, 7
Ventanas de resultados,
activar/desactivar, 288
Ventanas flotantes, 7
Vista,
Explorador, 85
Vista Authentic,
abrir archivo XML nuevo en, 227
abrir un documento XML en, 24
agregar nodos, 29
aplicar elementos, 29
ayudantes de entrada, 26
ayudantes de entrada en, 47
borrar elementos, 29
caracteres especiales en la, 32
edición de datos de BD en, 228
editar datos en una BD XML, 230
entidades en la, 32
formato de texto en, 41
generar documentos de salida a partir del archivo PXF, 236
iconos de la barra de herramientas, 41
iconos para tablas XML, 67
imprimir un documento XML desde la, 37
insertar entidades en la, 36
insertar nodos, 29
interfaz, 38
interfaz gráfica del usuario (IGU), 39
introducir datos en la, 32
introducir valores de atributo, 35
menú contextual, 26
menús contextuales, 52
mostrar etiquetas de marcado, 26
pegar como XML o como texto, 52
presentación de marcado en, 44
presentación del marcado en, 41
quitar nodos, 29
secciones CDATA en la, 32
tablas (SPS y XML), 61
tablas en la, 29
Tablas SPS, 62
tablas XML, 63
Tutorial, 22
uso de funciones importantes, 55
uso de tablas XML, 63
ventana principal de, 44
visualización de documentos, 44
Vista Explorador, 85, 237
actualizar el contenido, 241
adelante, 239
atrás, 238
dejar de cargar la página, 240
tamaño de la fuente, 242
ventana separada, 243
Vista predeterminada,
configuración de la ventana principal, 275
Vista previa de los atributos, 276
Visual Studio .Net,
diferencias con la versión independiente de Authentic Desktop, 129
y Authentic Desktop, 127
Volcar a cargar,
archivos cambiados, 150
VS .NET,
y el paquete de integración de Authentic, 128

W

Windows,
compatibilidad con productos de Altova, 726

X

XML,
Catálogo Oasis, 215
revisión ortográfica, 245
XMLData,
AppendChild, 597
EraseAllChildren, 598
EraseCurrentChild, 599
GetChild, 599
GetChildKind, 600
GetCurrentChild, 601
GetFirstChild, 601
GetNextChild, 602
HasChildren, 603
HasChildrenKind, 604
InsertChild, 604
IsSameNode, 605
Kind, 605
MayHaveChildren, 605
Name, 606
Parent, 606
TextValue, 607
XMLSPY, 140
ayuda, 21
complemento, 351
funciones, 21
registro del complemento, 352
XMLSpyLib, 373, 376
Application, 410
AuthenticDataTransfer, 432
AuthenticRange, 437
AuthenticView, 466
CodeGeneratorDlg, 483
DatabaseConnection, 489
Dialogs, 496
Document, 499
Documents, 528
DTDSchemaGeneratorDlg, 533
ElementList, 537
ElementListItem, 539
ExportSettings, 541
FileSelectionDlg, 544
GenerateSampleXMLDlg, 557
GridView, 562
ProjectItem, 582
SchemaDocumentationDlg, 566
SpyProject, 580
SpyProjectItems, 585
TextImportExportSettings, 587
XMLData, 596
XPath al nodo seleccionado, 39
XQuery,
pasar variables al documento XQuery, 222
XSLT,
procesador, 277

© 2015 Altova GmbH